

Optimization with Sparsity-Inducing Penalties

Francis Bach, Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski

▶ **To cite this version:**

Francis Bach, Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski. Optimization with Sparsity-Inducing Penalties. 2011. hal-00613125v1

HAL Id: hal-00613125

https://hal.science/hal-00613125v1

Preprint submitted on 2 Aug 2011 (v1), last revised 20 Nov 2011 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Optimization with Sparsity-Inducing Penalties

Francis Bach
INRIA - SIERRA Project-Team
23, avenue d'Italie
75013 Paris, France
francis.bach@inria.fr

Rodolphe Jenatton
INRIA - SIERRA Project-Team
23, avenue d'Italie
75013 Paris, France
rodolphe.jenatton@inria.fr

Julien Mairal
Department of Statistics
University of California
Berkeley, CA 94720, USA
julien@stat.berkeley.edu

Guillaume Obozinski
INRIA - SIERRA Project-Team
23, avenue d'Italie
75013 Paris, France
guillaume.obozinski@inria.fr

August 3, 2011

Abstract

Sparse estimation methods are aimed at using or obtaining parsimonious representations of data or models. They were first dedicated to linear variable selection but numerous extensions have now emerged such as structured sparsity or kernel selection. It turns out that many of the related estimation problems can be cast as convex optimization problems by regularizing the empirical risk with appropriate non-smooth norms. The goal of this paper is to present from a general perspective optimization tools and techniques dedicated to such sparsity-inducing penalties. We cover proximal methods, block-coordinate descent, reweighted ℓ_2-penalized techniques, working-set and homotopy methods, as well as non-convex formulations and extensions, and provide an extensive set of experiments to compare various algorithms from a computational point of view.

Contents

1 Introduction **1**

1.1 Notation 2

1.2 Loss functions 3

1.3 Sparsity-Inducing Norms 3

1.4 Optimization Tools 8

1.5 Multiple Kernel Learning 12

1.5.1 Variational formulation for sums of ℓ_2-norms 14

1.5.2 From the group Lasso to MKL 15

1.5.3 Variational formulations for subquadratic norms 16

1.5.4 Structured multiple kernel learning 18

2 Generic Methods **21**

3 Proximal Methods **23**

3.1 Principle of Proximal Methods 23

3.2 Algorithms 24

3.3 Computing the Proximal Operator 25

3.4 Proximal methods for structured MKL 27

4 (Block) Coordinate Descent Algorithms **30**

4.1 Coordinate descent for ℓ_1-regularization 30

4.2 Block-coordinate descent for ℓ_1/ℓ_q regularization 32

4.3 Block-coordinate descent for MKL 33

5 Reweighted-ℓ_2 Algorithms **34**

5.1 Quadratic variational formulation for general norms 35

6 Working-Set and Homotopy Methods **38**

6.1 Working-Set Techniques 38

6.2 Homotopy methods 40

7 Sparsity and Nonconvex Optimization **43**

7.1 Greedy Algorithms 43

7.2 DC-Programming, Reweighted-ℓ_1 Algorithms 45

7.3 Sparse Matrix Factorization and Dictionary Learning 46

7.4 Bayesian Methods 48

8 Quantitative Evaluation **49**

8.1 Speed Benchmarks for Lasso 50

8.2 Group-Sparsity for Multi-task Learning 51

8.3 Structured Sparsity 52

8.3.1 Denoising of natural image patches 52

8.3.2 Multi-class classification of cancer diagnosis. 53

8.3.3 General overlapping groups of variables 53

8.4 General Comments 54

9 Extensions **61**

10 Conclusions **62**

Chapter 1

Introduction

The principle of parsimony is central to many areas of science: the simplest explanation to a given phenomenon should be preferred over more complicated ones. In the context of machine learning, it takes the form of variable or feature selection, and it is commonly used in two situations. First, to make the model or the prediction more interpretable or computationally cheaper to use, i.e., even if the underlying problem is not sparse, one looks for the best sparse approximation. Second, sparsity can also be used given prior knowledge that the model should be sparse.

For variable selection in linear models, parsimony may be directly achieved by penalization of the empirical risk or the log-likelihood by the cardinality of the support[1] of the weight vector. However, this leads to hard combinatorial problems (see, e.g., [84, 116]). A traditional convex approximation of the problem is to replace the cardinality of the support by the ℓ_1-norm. Estimators may then be obtained as solutions of convex programs.

Casting sparse estimation as convex optimization problems has two main benefits: First, it leads to efficient estimation algorithms—and this paper focuses primarily on these. Second, it allows a fruitful theoretical analysis answering fundamental questions related to estimation consistency, prediction efficiency [19, 86] or model consistency [123, 133]. In particular, when the sparse model is assumed to be well-specified, regularization by the ℓ_1-norm is adapted to high-dimensional problems, where the number of variables to learn from may be exponential in the number of observations.

Reducing parsimony to finding the model of lowest cardinality turns out to be limiting, and *structured parsimony* [15, 55, 56, 53] has emerged as a natural extension, with applications to computer vision [31, 60, 53], text processing [59] or bioinformatics [55, 63]. Structured sparsity may be achieved by penalizing other functions than the cardinality of the support or regularizing by other norms than the ℓ_1-norm. In this paper, we focus primarily on norms which can be written as linear combinations of norms on subsets of variables, but we also consider traditional extensions such as multiple kernel learning and spectral norms on matrices (see Section 1.3 and 1.5). One main objective of this paper is to present methods

[1] We call the support the set of non-zeros

which are adapted to most sparsity-inducing norms with loss functions potentially beyond least-squares.

Finally, similar tools are used in other communities such as signal processing. While the objectives and the problem set-up are different, the resulting convex optimization problems are often similar, and most of the techniques reviewed in this paper also apply to sparse estimation problems in signal processing. Moreover, we consider in Section 7 non-convex formulations and extensions.

This paper aims at providing a general overview of the main optimization techniques that have emerged as most relevant and efficient for methods of variable selection based on sparsity inducing-norms. We survey and compare several algorithmic approaches as they apply to the ℓ_1-norm, group norms, but also to norms inducing structured sparsity and to general multiple kernel learning problems. We complement these by a presentation of some greedy and non-convex methods. Our presentation is essentially based on existing literature, but the process of constructing a general framework lead naturally to a couple of new results, connections and points of view.

This paper is organized as follows: We introduce some notation in Section 1.1, and present optimization problems related to sparse methods in Section 1.2. In Section 1.3, we present the various sparsity-inducing norms, while in Section 1.4, we review various optimization tools that will be needed throughout the paper. We then quickly present in Section 2 generic techniques that are not best suited to sparse methods. In subsequent sections, we present methods which are well adapted to regularized problems, namely proximal methods in Section 3, block coordinate descent in Section 4, reweighted ℓ_2-methods in Section 5, and working set and homotopy methods in Section 6. We review non-convex approaches such as greedy methods, DC programming and dictionary learning in Section 7. Finally, we provide quantitative evaluations of all of these methods in Section 8. Some of the material from this paper is taken from an earlier book chapter [12] and the dissertations of Rodolphe Jenatton and Julien Mairal.

1.1 Notation

Vectors are denoted by bold lower case letters and matrices by upper case ones. We define for $q \geq 1$ the ℓ_q-norm of a vector $\boldsymbol{x}$ in $\mathbb{R}^n$ as $\|\boldsymbol{x}\|_q := (\sum_{i=1}^n |\boldsymbol{x}_i|^q)^{1/q}$, where $\boldsymbol{x}_i$ denotes the i-th coordinate of $\boldsymbol{x}$, and $\|\boldsymbol{x}\|_\infty := \max_{i=1,\ldots,n} |\boldsymbol{x}_i| = \lim_{q\to\infty} \|\boldsymbol{x}\|_q$. We also define the ℓ_0-pseudo-norm as the number of nonzero elements in a vector:[2] $\|\boldsymbol{x}\|_0 := \#\{i \text{ s.t. } \boldsymbol{x}_i \neq 0\} = \lim_{q\to 0^+} (\sum_{i=1}^n |\boldsymbol{x}_i|^q)$. We consider the Frobenius norm of a matrix $\boldsymbol{X}$ in $\mathbb{R}^{m\times n}$: $\|\boldsymbol{X}\|_{\mathrm{F}} := (\sum_{i=1}^m \sum_{j=1}^n \boldsymbol{X}_{ij}^2)^{1/2}$, where $\boldsymbol{X}_{ij}$ denotes the entry of $\boldsymbol{X}$ at row i and column j. For an integer $n > 0$, and for any subset $J \subseteq \{1,\ldots,n\}$, we denote by $\boldsymbol{x}_J$ the vector of size $|J|$ containing the entries of a vector $\boldsymbol{x}$ in $\mathbb{R}^n$ indexed by J, and by $\boldsymbol{X}_J$ the matrix in $\mathbb{R}^{m\times|J|}$ containing the $|J|$ colums of a matrix $\boldsymbol{X}$ in $\mathbb{R}^{m\times n}$ indexed by J.

[2]Note that it would be more proper to write $\|\boldsymbol{x}\|_0^0$ instead of $\|\boldsymbol{x}\|_0$ to be consistent with the traditional notation $\|\boldsymbol{x}\|_q$. However, for the sake of simplicity, we will keep this notation unchanged in the rest of the paper.

1.2 Loss functions

We consider in this paper convex optimization problems of the form

$$\min_{\boldsymbol{w}\in\mathbb{R}^p} f(\boldsymbol{w}) + \lambda\Omega(\boldsymbol{w}), \tag{1.1}$$

where $f : \mathbb{R}^p \to \mathbb{R}$ is a convex differentiable function and $\Omega : \mathbb{R}^p \to \mathbb{R}$ is a sparsity-inducing—typically nonsmooth and non-Euclidean—norm.

In supervised learning, we predict outputs y in $\mathcal{Y}$ from observations $\boldsymbol{x}$ in $\mathcal{X}$; these observations are usually represented by p-dimensional vectors with $\mathcal{X} = \mathbb{R}^p$. In this supervised setting, f generally corresponds to the empirical risk of a loss function $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}_+$. More precisely, given n pairs of data points $\{(\boldsymbol{x}^{(i)}, y^{(i)}) \in \mathbb{R}^p \times \mathcal{Y};\ i = 1, \dots, n\}$, we have for linear models $f(\boldsymbol{w}) := \frac{1}{n}\sum_{i=1}^n \ell(y^{(i)}, \boldsymbol{w}^T\boldsymbol{x}^{(i)})$. Typical examples of differentiable loss functions are the square loss for least squares regression, i.e., $\ell(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$ with y in $\mathbb{R}$, and the logistic loss $\ell(y, \hat{y}) = \log(1 + e^{-y\hat{y}})$ for logistic regression, with y in $\{-1, 1\}$. We refer the readers to [109] for a more complete description of loss functions.

1.3 Sparsity-Inducing Norms

In this section, we present various norms as well as their main sparsity-inducing effects. These effects may be illustrated geometrically through the singularities of the corresponding unit balls (see Figure 1.4).

Sparsity through the ℓ_1-norm. When one knows *a priori* that the solutions $\boldsymbol{w}^\star$ of problem (1.1) should have a few non-zero coefficients, Ω is often chosen to be the ℓ_1-norm, i.e., $\Omega(\boldsymbol{w}) = \sum_{j=1}^p |\boldsymbol{w}_j|$. This leads for instance to the Lasso [114] with the square loss and to the ℓ_1-regularized logistic regression (see, for instance, [65, 110]) with the logistic loss. Regularizing by the ℓ_1-norm is known to induce sparsity in the sense that, a number of coefficients of $\boldsymbol{w}^\star$, depending on the strength of the regularization, will be *exactly* equal to zero.

Grouped ℓ_1-norms. In some situations, for example when encoding ordinal variables by binary dummy variables, the coefficients of $\boldsymbol{w}^\star$ are naturally partitioned in subsets, or *groups*, of variables. It is then natural to select or remove *simultaneously* all the variables forming a group. A regularization norm exploiting explicitly this group structure can be shown to improve the prediction performance and/or interpretability of the learned models [52, 71, 92, 102, 120, 131]. Such a norm might for instance take the form

$$\Omega(\boldsymbol{w}) := \sum_{g\in\mathcal{G}} d_g \|\boldsymbol{w}_g\|_2, \tag{1.2}$$

where $\mathcal{G}$ is a partition of $\{1, \dots, p\}$, $(d_g)_{g\in\mathcal{G}}$ are some positive weights, and $\boldsymbol{w}_g$ denotes the vector in $\mathbb{R}^{|g|}$ recording the coefficients of $\boldsymbol{w}$ indexed by g in $\mathcal{G}$. Without loss of generality

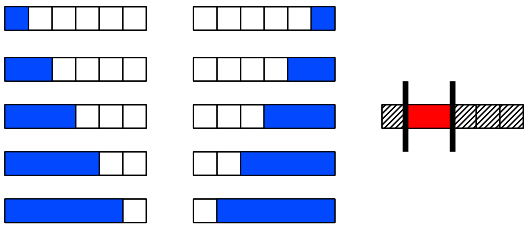


Figure 1.1: (Left) The set of blue groups to penalize in order to select contiguous patterns in a sequence. (Right) In red, an example of such a nonzero pattern with its corresponding zero pattern (hatched area).

we may assume all weights $(d_g)_{g\in\mathcal{G}}$ to be equal to one (when $\mathcal{G}$ is a partition, we can rescale the values of $\boldsymbol{w}$ appropriately). As defined in Eq. (1.2), Ω is known as a mixed ℓ_1/ℓ_2-norm. It behaves like an ℓ_1-norm on the vector $(\|\boldsymbol{w}_g\|_2)_{g\in\mathcal{G}}$ in $\mathbb{R}^{|\mathcal{G}|}$, and therefore, Ω induces group sparsity. In other words, each $\|\boldsymbol{w}_g\|_2$, and equivalently each $\boldsymbol{w}_g$, is encouraged to be set to zero. On the other hand, within the groups g in $\mathcal{G}$, the ℓ_2-norm does not promote sparsity. Combined with the square loss, it leads to the group Lasso formulation [131]. Note that when $\mathcal{G}$ is the set of singletons, we retrieve the ℓ_1-norm. More general mixed ℓ_1/ℓ_q-norms for $q > 1$ are also used in the literature [132]:

$$\Omega(\boldsymbol{w}) = \sum_{g\in\mathcal{G}} \|\boldsymbol{w}_g\|_q := \sum_{g\in\mathcal{G}} d_g \Big\{ \sum_{j\in g} |\boldsymbol{w}_j|^q \Big\}^{1/q}.$$

In practice though, the ℓ_1/ℓ_2- and ℓ_1/ℓ_∞-settings remain the most popular ones. Note that using ℓ_∞-norms may have the undesired effect to favor solutions $\boldsymbol{w}$ with many components of equal magnitude (due to the extra non-differentiabilities away from zero).

Grouped ℓ_1-norms are typically used when extra-knowledge is available regarding an appropriate partition, in particular in the presence of categorical variables with orthogonal encoding [102], for multi-task learning where joint variable selection is desired [92], and for multiple kernel learning (see Section 1.5).

Norms for overlapping groups: a direct formulation. In an attempt to better encode structural links between variables at play (e.g., spatial or hierarchical links related to the physics of the problem at hand), recent research has explored the setting where $\mathcal{G}$ in Eq. (1.2) can contain groups of variables that *overlap* [9, 55, 56, 63, 104, 132]. In this case, Ω is still a norm, and it yields sparsity in the form of specific patterns of variables. More precisely, the solutions $\boldsymbol{w}^\star$ of problem (1.1) can be shown to have a set of zero coefficients, or simply *zero pattern*, that corresponds to a union of some groups g in $\mathcal{G}$ [56]. This property makes it possible to control the sparsity patterns of $\boldsymbol{w}^\star$ by appropriately defining the groups in $\mathcal{G}$. Note that here the weights d_g should not be taken equal to one (see, [56] for more details). This form of *structured sparsity* has notably proven to be useful in various contexts, which we now illustrate through concrete examples:

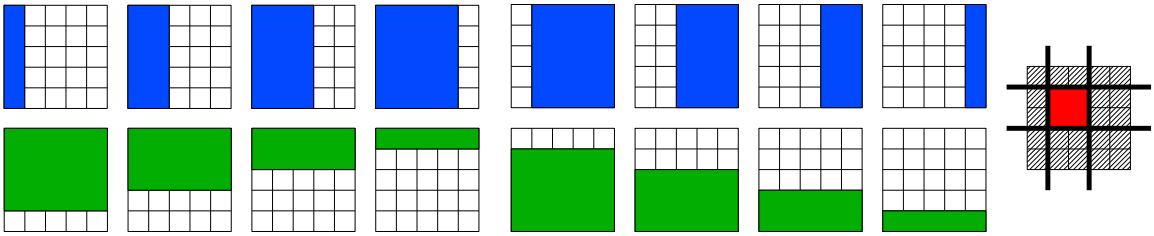


Figure 1.2: Vertical and horizontal groups: (Left) the set of blue and green groups to penalize in order to select rectangles. (Right) In red, an example of nonzero pattern recovered in this setting, with its corresponding zero pattern (hatched area).

- **One-dimensional Sequence:** Given p variables organized in a sequence, if we want to select only contiguous nonzero patterns, we represent in Figure 1.1 the set of groups $\mathcal{G}$ to consider. In this case, we have $|\mathcal{G}| = O(p)$. Imposing the contiguity of the nonzero patterns is for instance relevant in the context of time series, or for the diagnosis of tumors, based on the profiles of arrayCGH [98]. Indeed, because of the specific spatial organization of bacterial artificial chromosomes along the genome, the set of discriminative features is expected to have specific contiguous patterns.

- **Two-dimensional Grid:** In the same way, assume now the p variables are organized on a two-dimensional grid. If we want the possible nonzero patterns $\mathcal{P}$ to be the set of all rectangles on this grid, the appropriate groups $\mathcal{G}$ to consider can be shown (see [56]) to be those represented in Figure 1.2. In this setting, we have $|\mathcal{G}| = O(\sqrt{p})$. Sparsity-inducing regularizations built upon such group structures have resulted in good performances for background subtraction [53, 75, 77], topographic dictionary learning [62, 77], wavelet-based denoising [97], and for face recognition with corruption by occlusions [60].

- **Hierarchical Structure:** A third interesting example assumes that the variables have a hierarchical structure. Specifically, we consider that the p variables correspond to the nodes of tree $\mathcal{T}$ (or a forest of trees). Moreover, we assume that we want to select the variables according to a certain order: a feature can be select only if all its ancestors in $\mathcal{T}$ are already selected. This hierarchical rule can be shown to lead to the family of groups displayed on Figure 1.3.

  This resulting penalty was first used in [132]; since then, this type of groups has led to numerous applications, for instance, wavelet-based denoising [15, 53, 58, 132], hierarchical dictionary learning for both topic modeling and image restoration [58, 59], log-linear models for the selection of potential orders [104], bioinformatics, to exploit the tree structure of gene networks for multi-task regression [63], and multi-scale mining of fMRI data for the prediction of some cognitive task [57]. More recently, this hierarchical penalty was proved to be efficient for template selection in natural lan-

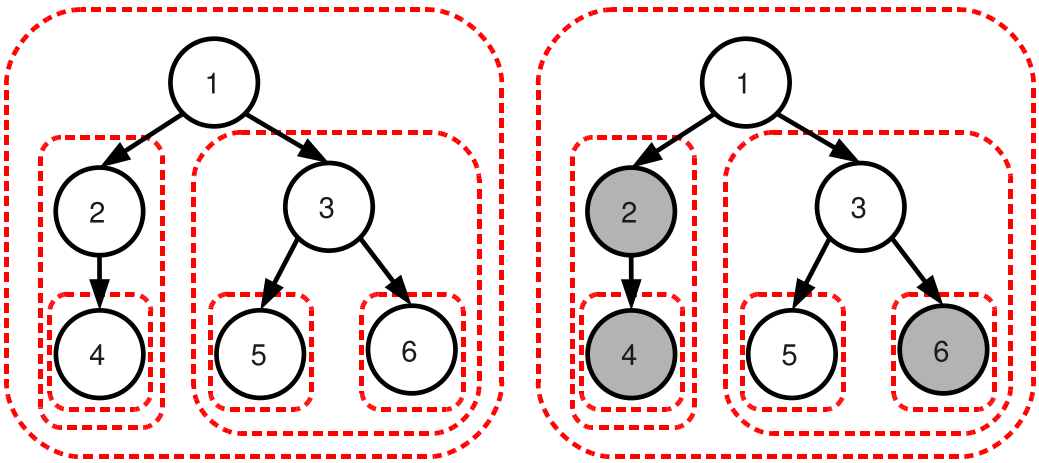


Figure 1.3: Left: example of a tree-structured set of groups $\mathcal{G}$ (dashed contours in red), corresponding to a tree $\mathcal{T}$ with $p = 6$ nodes represented by black circles. Right: example of a sparsity pattern induced by the tree-structured norm corresponding to $\mathcal{G}$: the groups $\{2, 4\}, \{4\}$ and $\{6\}$ are set to zero, so that the corresponding nodes (in gray) that form subtrees of $\mathcal{T}$ are removed. The remaining nonzero variables $\{1, 3, 5\}$ form a rooted and connected subtree of $\mathcal{T}$. This sparsity pattern obeys the following equivalent rules: (i) if a node is selected, the same goes for all its ancestors; (ii) if a node is not selected, then its descendant are not selected.

guage processing [80].

- **Extensions:** The possible choices for the sets of groups $\mathcal{G}$ are not limited to the aforementioned examples. More complicated topologies can be considered, for instance, three-dimensional spaces discretized in cubes or spherical volumes discretized in slices; for instance, see [121] for an application to neuroimaging that pursues this idea. Moreover, directed acyclic graphs that extends the trees presented in Figure 1.3 have notably proven to be useful in the context of hierarchical variable selection [132, 9, 104],

Norms for overlapping groups: a latent variable formulation. The family of norms defined in Eq. (1.2) is adapted to *intersection-closed* sets of nonzero patterns. However, some applications exhibit structures that can be more naturally modelled by *union-closed* families of supports. This idea was developed by [55] who, given a set of groups $\mathcal{G}$, introduced the following norm

$$\Omega_{\text{union}}(\boldsymbol{w}) := \min_{\boldsymbol{v} \in \mathbb{R}^{p \times |\mathcal{G}|}} \sum_{g \in \mathcal{G}} d_g \|\boldsymbol{v}^g\|_2 \quad \text{such that} \quad \left\{ \begin{array}{l} \sum_{g \in \mathcal{G}} \boldsymbol{v}^g = \boldsymbol{w}, \\ \forall g \in \mathcal{G},\ \boldsymbol{v}^g_j = 0 \text{ if } j \notin g. \end{array} \right. \tag{1.3}$$

The norm defined above provides a different generalization of the ℓ_1/ℓ_2 norm to the case of overlapping groups that the norm presented above. The choice of the weights d_g is crucial in this setting as well. It may be interpreted as duplicating the variables that belong to several

groups and penalizing the norms of all duplicates using an ℓ_1/ℓ_2-norm. Interestingly, it can be interpreted as a convex relaxation of a non-convex penalty introduced in [53], which encourages similar sparsity patterns.

Multiple kernel learning. For most of the sparsity-inducing terms described in this paper, we may replace real variables and their absolute values by pre-defined groups of variables with their Euclidean norms (we have already seen such examples with ℓ_1/ℓ_2-norms), or more generally, by members of reproducing kernel Hilbert spaces. As shown in Section 1.5, most of the tools that we present in this paper are applicable to this case as well, through appropriate modifications and borrowing of tools from kernel methods, and have applications in particular in multiple kernel learning. Note that this extension requires tools from convex analysis presented in Section 1.4.

Trace norm. Given learning problem on matrices, such as matrix completion, the rank plays a similar role than the cardinality of the support for vectors. Indeed, the rank of a matrix $\mathbf{M}$ may be seen as the number of non-zero singular values of $\mathbf{M}$. The rank of $\mathbf{M}$ however is not a continuous function of $\mathbf{M}$, and, following the convex relaxation of the ℓ_0-pseudo-norm into the ℓ_1-norm, we may relax the rank of $\mathbf{M}$ into the sum of its singular values, which happens to be a norm, and is often referred to as the trace norm or nuclear norm of $\mathbf{M}$, and which we denote $\|\mathbf{M}\|_*$. As shown in this paper, many of the tools designed for the ℓ_1-norm may be extended to the trace norm.

Using the trace norm as a convex surrogate for rank has many applications in control theory [44], matrix completion [1, 112], multi-task learning [95], or multi-label classification [4], where low-rank priors are adapted.

Sparsity-inducing properties: a geometrical intuition. Although we consider in Eq. (1.1) a regularized formulation, we could equivalently focus on a *constrained* problem, that is,

$$\min_{\boldsymbol{w}\in\mathbb{R}^p} f(\boldsymbol{w}) \quad \text{such that} \quad \Omega(\boldsymbol{w}) \leq \mu, \tag{1.4}$$

for some $\mu \in \mathbb{R}_+$. The set of solutions of Eq. (1.4) parameterized by μ is the same as that of Eq. (1.1), as described by some value of λ_μ depending on μ (e.g., see Section 3.2 in [20]). At optimality, the gradient of f evaluated at any solution $\hat{\boldsymbol{w}}$ of (1.4) is known to belong to the normal cone to $\mathcal{B} = \{\boldsymbol{w} \in \mathbb{R}^p;\ \Omega(\boldsymbol{w}) \leq \mu\}$ at $\hat{\boldsymbol{w}}$ [20]. In other words, for sufficiently small values of μ, i.e., so that the constraint is active, the level set of f for the value $f(\hat{\boldsymbol{w}})$ is tangent to $\mathcal{B}$.

As a consequence, the geometry of the ball $\mathcal{B}$ is directly related to the properties of the solutions $\hat{\boldsymbol{w}}$. If Ω is taken to be the ℓ_2-norm, then the resulting ball $\mathcal{B}$ is the standard, isotropic, "round" ball that does not favor any specific direction of the space. On the other hand, when Ω is the ℓ_1-norm, $\mathcal{B}$ corresponds to a diamond-shaped pattern in two dimensions, and to a pyramid in three dimensions. In particular, $\mathcal{B}$ is anisotropic and exhibits some singular points due to the non-smoothness of Ω. Moreover, these singular points are located along the axis of $\mathbb{R}^p$, so that if the level set of f happens to be tangent

at one of those points, sparse solutions are obtained. We display in Figure 1.4 the balls $\mathcal{B}$ for the ℓ_1-, ℓ_2-, and two different grouped ℓ_1/ℓ_2-norms.

Extensions. The design of sparsity-inducing norms is an active field of research and similar tools than the ones we present in this can be derived for other norms. As shown in Section 3, computing the proximal operator readily leads to efficient algorithms, and for the extensions we present below, these operators can be efficiently computed.

In order to impose prior knowledge on the support of predictor, the norms based on overlapping ℓ_1/ℓ_∞-norms can be shown to be convex relaxations of submodular functions of the support, and further ties can be made between convex optimization and combinatorial optimization (see [10] for more details).

Moreover, similar developments may be carried through for norms that try to enforce that the predictors have many equal components and that the resulting clusters have specific shapes, e.g., contiguous in a pre-defined order, see, e.g., [11, 32, 75, 122, 115] and references therein.

1.4 Optimization Tools

The tools used in this paper are relatively basic and should be accessible to a broad audience. Most of them can be found in classical books on convex optimization [18, 20, 25, 91], but for self-containedness, we present here a few of them related to non-smooth unconstrained optimization. In particular, these tools allow the derivation of rigorous approximate optimality conditions based on duality gaps (instead of relying on weak stopping criteria based on small changes or low-norm gradients).

Subgradients. Given a convex function $g : \mathbb{R}^p \to \mathbb{R}$ and a vector $\boldsymbol{w}$ in $\mathbb{R}^p$, let us define the *subdifferential* of g at $\boldsymbol{w}$ as

$$\partial g(\boldsymbol{w}) := \{\boldsymbol{z} \in \mathbb{R}^p \mid g(\boldsymbol{w}) + \boldsymbol{z}^T(\boldsymbol{w}' - \boldsymbol{w}) \leq g(\boldsymbol{w}') \text{ for all vectors } \boldsymbol{w}' \in \mathbb{R}^p\}.$$

The elements of $\partial g(\boldsymbol{w})$ are called the *subgradients* of g at $\boldsymbol{w}$. Note that all convex functions defined on $\mathbb{R}^p$ have non-empty subdifferentials at every point. This definition admits a clear geometric interpretation: any subgradient $\boldsymbol{z}$ in $\partial g(\boldsymbol{w})$ defines an affine function $\boldsymbol{w}' \mapsto g(\boldsymbol{w}) + \boldsymbol{z}^T(\boldsymbol{w}' - \boldsymbol{w})$ which is tangent to the graph of the function g (because of the convexity of g, it is a lower-bounding tangent). Moreover, there is a bijection (one-to-one correspondence) between such "tangent affine functions" and the subgradients, as illustrated in Figure 1.5. Subdifferentials are useful for studying nonsmooth optimization problems because of the following proposition (whose proof is straightforward from the definition):

Proposition 1.1 (Subgradients at Optimality)
For any convex function $g : \mathbb{R}^p \to \mathbb{R}$, a point $\boldsymbol{w}$ in $\mathbb{R}^p$ is a global minimum of g if and only if the condition $0 \in \partial g(\boldsymbol{w})$ holds.

Note that the concept of subdifferential is mainly useful for nonsmooth functions. If g is differentiable at $\boldsymbol{w}$, the set $\partial g(\boldsymbol{w})$ is indeed the singleton $\{\nabla g(\boldsymbol{w})\}$, and the condition $0 \in \partial g(\boldsymbol{w})$ reduces to the classical first-order optimality condition $\nabla g(\boldsymbol{w}) = 0$. As a simple example, let us consider the following optimization problem

$$\min_{w \in \mathbb{R}} \frac{1}{2}(x - w)^2 + \lambda |w|.$$

Applying the previous proposition and noting that the subdifferential $\partial|\cdot|$ is $\{+1\}$ for $w > 0$, $\{-1\}$ for $w < 0$ and $[-1, 1]$ for $w = 0$, it is easy to show that the unique solution admits a closed form called the *soft-thresholding* operator, following a terminology introduced by [38]; it can be written

$$w^\star = \begin{cases} 0 & \text{if } |x| \leq \lambda \\ (1 - \frac{\lambda}{|x|})x & \text{otherwise,} \end{cases} \tag{1.5}$$

or equivalently $w^\star = \text{sign}(x)(|x| - \lambda)_+$. This operator is a core component of many optimization techniques for sparse methods, as we shall see later. Its counterpart for non-convex optimization problems is the hard-thresholding operator. Both of them are presented in Figure 1.6. Note that similar developments could be carried through using directional derivatives instead of sugradients (see, e.g., [20]).

Dual Norm and Optimality Conditions. The next concept we introduce is the dual norm, which is important to study sparsity-inducing regularizations [9, 56, 86]. It notably arises in the analysis of estimation bounds [86], and in the design of working-set strategies as will be shown in Section 6.1. The dual norm Ω^* of the norm Ω is defined for any vector $\boldsymbol{z}$ in $\mathbb{R}^p$ by

$$\Omega^*(\boldsymbol{z}) := \max_{\boldsymbol{w} \in \mathbb{R}^p} \boldsymbol{z}^T \boldsymbol{w} \text{ such that } \Omega(\boldsymbol{w}) \leq 1. \tag{1.6}$$

Moreover, the dual norm of Ω^* is Ω itself, and as a consequence, the formula above holds also if the roles of Ω and Ω^* are exchanged. It is easy to show that in the case of an ℓ_q-norm, $q \in [1; +\infty]$, the dual norm is the $\ell_{q'}$-norm, with q' in $[1; +\infty]$ such that $\frac{1}{q} + \frac{1}{q'} = 1$. In particular, the ℓ_1- and ℓ_∞-norms are dual to each other, and the ℓ_2-norm is self-dual (dual to itself).

The dual norm plays a direct role in computing optimality conditions of sparse regularized problems. By applying Proposition 1.1 to Eq. (1.1), a little calculation shows that a vector $\boldsymbol{w}$ in $\mathbb{R}^p$ is optimal for Eq. (1.1) if and only if $-\frac{1}{\lambda}\nabla f(\boldsymbol{w}) \in \partial\Omega(\boldsymbol{w})$ with

$$\partial\Omega(\boldsymbol{w}) = \begin{cases} \{\boldsymbol{z} \in \mathbb{R}^p;\ \Omega^*(\boldsymbol{z}) \leq 1\} \text{ if } \boldsymbol{w} = 0, \\ \{\boldsymbol{z} \in \mathbb{R}^p;\ \Omega^*(\boldsymbol{z}) \leq 1 \text{ and } \boldsymbol{z}^T\boldsymbol{w} = \Omega(\boldsymbol{w})\} \text{ otherwise.} \end{cases} \tag{1.7}$$

As a consequence, the vector $\mathbf{0}$ is solution if and only if $\Omega^*\big(\nabla f(\mathbf{0})\big) \leq \lambda$. Note that this shows that for all λ larger than $\Omega^*\big(\nabla f(\mathbf{0})\big)$, $\boldsymbol{w} = \mathbf{0}$ is a solution of the regularized optimization problem (hence this value is the start of the non-trivial regularization path). A proof of the equality in Eq. (1.7) is postponed to the next section, in Remark 1.1.

These general optimality conditions can be specified to the Lasso problem [114], also known as basis pursuit [33]:

$$\min_{\boldsymbol{w}\in\mathbb{R}^p} \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda\|\boldsymbol{w}\|_1, \tag{1.8}$$

where $\boldsymbol{y}$ is in $\mathbb{R}^n$, and $\boldsymbol{X}$ is a design matrix in $\mathbb{R}^{n\times p}$. From Eq. (1.7) and since the ℓ_∞-norm is the dual of the ℓ_1-norm we obtain that necessary and sufficient optimality conditions are

$$\forall j = 1,\dots,p, \quad \left\{ \begin{array}{rcll} |\boldsymbol{X}_j^T(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w})| & \leq & \lambda & \text{if } \boldsymbol{w}_j = 0 \\ \boldsymbol{X}_j^T(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w}) & = & \lambda\operatorname{sgn}(\boldsymbol{w}_j) & \text{if } \boldsymbol{w}_j \neq 0, \end{array} \right. \tag{1.9}$$

where $\boldsymbol{X}_j$ denotes the j-th column of $\boldsymbol{X}$, and $\boldsymbol{w}_j$ the j-th entry of $\boldsymbol{w}$. As we will see in Section 6.2, it is possible to derive from these conditions interesting properties of the Lasso, as well as efficient algorithms for solving it. We have presented a useful duality tool for norms. More generally, there exists a related concept for convex functions, which we now introduce.

Fenchel Conjugate and Duality Gaps. Let us denote by f^* the Fenchel conjugate of f [101], defined by

$$f^*(\boldsymbol{z}) := \sup_{\boldsymbol{w}\in\mathbb{R}^p} [\boldsymbol{z}^T\boldsymbol{w} - f(\boldsymbol{w})].$$

Fenchel conjugates are particulary useful to derive dual problems and duality gaps[3]. Under mild conditions, the conjugate of the conjugate of a convex function is itself, leading to the following representation of f as a maximum of affine functions:

$$f(\boldsymbol{w}) = \sup_{\boldsymbol{z}\in\mathbb{R}^p} [\boldsymbol{z}^T\boldsymbol{w} - f^*(\boldsymbol{z})].$$

In the context of this tutorial, it is notably useful to specify the expression of the conjugate of a norm. Perhaps surprisingly and misleadingly, the conjugate of a norm is not equal to its dual norm, but corresponds instead to the indicator function of the unit ball of its dual norm. More formally, let us introduce the indicator function ι_{Ω^*} such that $\iota_{\Omega^*}(\boldsymbol{z})$ is equal to 0 if $\Omega^*(\boldsymbol{z}) \leq 1$ and $+\infty$ otherwise. Then, we have the equality $\iota_{\Omega^*} = \sup_{\boldsymbol{w}\in\mathbb{R}^p}[\boldsymbol{z}^T\boldsymbol{w} - \Omega(\boldsymbol{w})]$. This result is well-known and appears in several text books (e.g., see Example 3.26 in [25]); for the sake of completeness, we give below of proof of this equality:

Proposition 1.2 (Fenchel conjugate of a norm) *Let Ω be a norm on $\mathbb{R}^p$. The following equality holds for any $\boldsymbol{z} \in \mathbb{R}^p$*

$$\sup_{\boldsymbol{w}\in\mathbb{R}^p} [\boldsymbol{z}^T\boldsymbol{w} - \Omega(\boldsymbol{w})] = \begin{cases} 0 & \text{if } \Omega^*(\boldsymbol{z}) \leq 1 \\ +\infty & \text{otherwise.} \end{cases}$$

Proof On the one hand, assume that the dual norm of $\boldsymbol{z}$ is greater than one, that is, $\Omega^*(\boldsymbol{z}) > 1$. According to the definition of the dual norm (see Eq. (1.6)), and since the supremum is taken over the compact set $\{\boldsymbol{w} \in \mathbb{R}^p;\ \Omega(\boldsymbol{w}) \leq 1\}$, there exists a vector $\boldsymbol{w}$ in

[3]For many of our norms, *conic* duality tools would suffice (see, e.g., [25]).

this ball such that $\Omega^*(\boldsymbol{z}) = \boldsymbol{z}^\top \boldsymbol{w} > 1$. For any scalar $t \geq 0$, consider $\boldsymbol{v} = t\boldsymbol{w}$ and notice that

$$\boldsymbol{z}^\top \boldsymbol{v} - \Omega(\boldsymbol{v}) = t[\boldsymbol{z}^\top \boldsymbol{w} - \Omega(\boldsymbol{w})] \geq t,$$

which shows that when $\Omega^*(\boldsymbol{z}) > 1$, the Fenchel conjugate is unbounded.

Now, assume that $\Omega^*(\boldsymbol{z}) \leq 1$. By applying the generalized Cauchy-Schwartz's inequality, we obtain for any $\boldsymbol{w}$

$$\boldsymbol{z}^\top \boldsymbol{w} - \Omega(\boldsymbol{w}) \leq \Omega^*(\boldsymbol{z})\Omega(\boldsymbol{w}) - \Omega(\boldsymbol{w}) \leq 0.$$

Equality holds for $\boldsymbol{w} = \boldsymbol{0}$, and the conclusion follows. ■

Remark 1.1 *With Proposition 1.2 in place, we can formally (and easily) prove the relationship in Eq. (1.7) that explicits the subdifferential of a norm. Based on Proposition 1.2, we indeed know that the conjuguate of Ω is ι_{Ω^*}. Applying the Fenchel-Young inequality (see Proposition 3.3.4 in [20]), we have*

$$\boldsymbol{z} \in \partial\Omega(\boldsymbol{w}) \Leftrightarrow \Big[\boldsymbol{z}^\top \boldsymbol{w} = \Omega(\boldsymbol{w}) + \iota_{\Omega^*}(\boldsymbol{z})\Big],$$

which leads to the desired conclusion.

For many objective functions, the Fenchel conjugate admits closed forms, and can therefore be computed efficiently [20]. Then, it is possible to derive a duality gap for problem (1.1) from standard Fenchel duality arguments (see [20]), as shown in the following proposition:

Proposition 1.3 (Duality for Problem (1.1))
If f^* and Ω^* are respectively the Fenchel conjugate of a convex and differentiable function f and the dual norm of Ω, then we have

$$\max_{\boldsymbol{z} \in \mathbb{R}^p : \Omega^*(\boldsymbol{z}) \leq \lambda} -f^*(\boldsymbol{z}) \quad \leq \quad \min_{\boldsymbol{w} \in \mathbb{R}^p} f(\boldsymbol{w}) + \lambda\Omega(\boldsymbol{w}). \tag{1.10}$$

Moreover, equality holds as soon as the domain of f has non-empty interior.

Proof *This result is a specific instance of Theorem 3.3.5 in [20]. In particular, we use the fact that the conjugate of a norm Ω is the indicator function ι_{Ω^*} of the unit ball of the dual norm Ω^* (see Proposition 1.2).*

If $\boldsymbol{w}^\star$ is a solution of Eq. (1.1), and $\boldsymbol{w}, \boldsymbol{z}$ in $\mathbb{R}^p$ are such that $\Omega^*(\boldsymbol{z}) \leq \lambda$, this proposition implies that we have

$$f(\boldsymbol{w}) + \lambda\Omega(\boldsymbol{w}) \geq f(\boldsymbol{w}^\star) + \lambda\Omega(\boldsymbol{w}^\star) \geq -f^*(\boldsymbol{z}). \tag{1.11}$$

The difference between the left and right term of Eq. (1.11) is called a duality gap. It represents the difference between the value of the primal objective function $f(\boldsymbol{w}) + \lambda\Omega(\boldsymbol{w})$ and a dual objective function $-f^*(\boldsymbol{z})$, where $\boldsymbol{z}$ is a dual variable. The proposition says that the duality gap for a pair of optima $\boldsymbol{w}^\star$ and $\boldsymbol{z}^\star$ of the primal and dual problem is equal to 0. When the optimal duality gap is zero one says that *strong duality* holds.

Duality gaps are important in convex optimization because they provide an upper bound on the difference between the current value of an objective function and the optimal value, which makes it possible to set proper stopping criteria for iterative optimization algorithms. Given a current iterate $\boldsymbol{w}$, computing a duality gap requires choosing a "good" value for $\boldsymbol{z}$ (and in particular a feasible one). Given that at optimality, $\boldsymbol{z}(\boldsymbol{w}^\star) = \nabla f(\boldsymbol{w}^\star)$ is the unique solution to the dual problem, a natural choice of dual variable is $\boldsymbol{z} = \min\big(1, \frac{\lambda}{\Omega^*(\nabla f(\boldsymbol{w}))}\big)\nabla f(\boldsymbol{w})$, which reduces to $\boldsymbol{z}(\boldsymbol{w}^\star)$ at the optimum and therefore yields a zero duality gap at optimality.

Note that in most formulations that we will consider, the function f is of the form $f(\boldsymbol{w}) = \psi(\boldsymbol{X}\boldsymbol{w})$ with $\psi : \mathbb{R}^n \to \mathbb{R}$ and $\boldsymbol{X}$ a design matrix; typically, the Fenchel conjugate of ψ is easy to compute[4] while the design matrix $\boldsymbol{X}$ makes it hard[5] to compute f^*. In that case, Eq. (1.1) can be rewritten as

$$\min_{\boldsymbol{w}\in\mathbb{R}^p, \boldsymbol{u}\in\mathbb{R}^n} \quad \psi(\boldsymbol{u}) + \lambda\,\Omega(\boldsymbol{w}) \qquad \text{s.t. } \boldsymbol{u} = \boldsymbol{X}\boldsymbol{w}, \tag{1.12}$$

and equivalently as the optimization of the Lagrangian

$$\min_{\boldsymbol{w}\in\mathbb{R}^p, \boldsymbol{u}\in\mathbb{R}^n} \quad \max_{\boldsymbol{\alpha}\in\mathbb{R}^n} \quad \big(\psi(\boldsymbol{u}) - \lambda\boldsymbol{\alpha}^T\boldsymbol{u}\big) + \lambda\,\big(\Omega(\boldsymbol{w}) + \boldsymbol{\alpha}^T\boldsymbol{X}\boldsymbol{w}\big), \tag{1.13}$$

which is obtained by introducing the Lagrange multiplier $\boldsymbol{\alpha}$. The corresponding Fenchel dual[6] is then

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^n} \quad -\psi^*(\lambda\boldsymbol{\alpha}) \quad \text{such that} \quad \Omega^*(\boldsymbol{X}^T\boldsymbol{\alpha}) \leq 1, \tag{1.14}$$

which does not require any inversion of $\boldsymbol{X}^T\boldsymbol{X}$ (which would be required for computing the Fenchel conjugate of f). Thus, given a candidate $\boldsymbol{w}$, we consider $\boldsymbol{\alpha} = \min\big(1, \frac{\lambda}{\Omega^*(\boldsymbol{X}^T\psi'(\boldsymbol{X}\boldsymbol{w}))}\big)\psi'(\boldsymbol{X}\boldsymbol{w})$, and can get an upper bound on optimality using primal (1.12) and dual (1.14) problems. Concrete examples of such duality gaps for various sparse regularized problems are presented in appendix D of [74], and are implemented in the open-source software SPAMS[7], which we have used in the experimental section of this paper.

1.5 Multiple Kernel Learning

A seemingly unrelated problem in machine learning, the problem of *multiple kernel learning* is in fact intimately connected with sparsity-inducing norms by duality. It actually corresponds to the most natural extension of sparsity to reproducing kernel Hilbert spaces. We will show that for a large class of norms and, among them, many sparsity-inducing norms, there exists for each of them a corresponding multiple kernel learning scheme, and, vice versa, each multiple kernel learning scheme defines a new norm.

[4] For the least-squares loss with output vector $\boldsymbol{y} \in \mathbb{R}^n$, we have $\psi(\boldsymbol{u}) = \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{u}\|_2^2$ and $\psi^*(\boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{\beta}\|_2^2 + \boldsymbol{\beta}^T\boldsymbol{y}$. For the logistic loss, we have $\psi(\boldsymbol{u}) = \sum_{i=1}^n \log(1 + \exp(-\boldsymbol{y}_i\boldsymbol{u}_i))$ and $\psi^*(\boldsymbol{\beta}) = \sum_{i=1}^n (1 + \boldsymbol{\beta}_i\boldsymbol{y}_i)\log(1 + \boldsymbol{\beta}_i\boldsymbol{y}_i) - \boldsymbol{\beta}_i\boldsymbol{y}_i\log(-\boldsymbol{\beta}_i\boldsymbol{y}_i)$ if $\forall i,\ -\boldsymbol{\beta}_i\boldsymbol{y}_i \in [0, 1]$ and $+\infty$ otherwise.

[5] It would require to compute the pseudo-inverse of $\boldsymbol{X}$.

[6] Fenchel conjugacy naturally extends to this case; see Theorem 3.3.5 in [20] for more details.

[7] `http://www.di.ens.fr/willow/SPAMS/`

The problem of kernel learning is a priori quite unrelated with parsimony. It emerges as a consequence of a convexity property of the so-called "kernel trick", which we now describe. Consider a learning problem with $f(\boldsymbol{w}) = \psi(\boldsymbol{X}\boldsymbol{w})$, but regularized this time by the square of the norm

$$\min_{\boldsymbol{w}\in\mathbb{R}^p} f(\boldsymbol{w}) + \frac{\lambda}{2}\Omega(\boldsymbol{w})^2. \tag{1.15}$$

As in Eq. (1.12) we can introduce linear constraint

$$\min_{\boldsymbol{u}\in\mathbb{R}^n, \boldsymbol{w}\in\mathbb{R}^p} \psi(\boldsymbol{u}) + \frac{\lambda}{2}\Omega(\boldsymbol{w})^2 \quad \text{s.t.} \quad \boldsymbol{u} = \boldsymbol{X}\boldsymbol{w}, \tag{1.16}$$

and reformulate the problem as the saddle point problem

$$\min_{\boldsymbol{u}\in\mathbb{R}^n, \boldsymbol{w}\in\mathbb{R}^p} \max_{\alpha\in\mathbb{R}^n} \psi(\boldsymbol{u}) + \frac{\lambda}{2}\Omega(\boldsymbol{w})^2 - \lambda\boldsymbol{\alpha}^\top(\boldsymbol{u} - \boldsymbol{X}\boldsymbol{w}). \tag{1.17}$$

Since the primal problem (1.16) is a convex problem with feasible linear constraints, it satisfies Slater's qualification conditions and the order of maximization and minimization can be exchanged

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^n} \min_{\boldsymbol{u}\in\mathbb{R}^n, \boldsymbol{w}\in\mathbb{R}^p} (\psi(\boldsymbol{u}) - \lambda\boldsymbol{\alpha}^\top\boldsymbol{u}) + \lambda\big(\frac{1}{2}\Omega(\boldsymbol{w})^2 + \boldsymbol{\alpha}^\top\boldsymbol{X}\boldsymbol{w}\big). \tag{1.18}$$

Now, the minimization in $\boldsymbol{u}$ and $\boldsymbol{w}$ can be performed independently. One property of norms is that the Fenchel conjugate of $\boldsymbol{w} \mapsto \frac{1}{2}\Omega(\boldsymbol{w})^2$ is $\boldsymbol{\kappa} \mapsto \frac{1}{2}\Omega^*(\boldsymbol{\kappa})^2$; this can be easily verified by finding the vector $\boldsymbol{w}$ achieving equality in the sequence of inequalities $\boldsymbol{\kappa}^\top\boldsymbol{w} \leq \Omega(\boldsymbol{w})\,\Omega^*(\boldsymbol{\kappa}) \leq \frac{1}{2}\big[\Omega(\boldsymbol{w})^2 + \Omega^*(\boldsymbol{\kappa})^2\big]$. As a consequence, the dual optimization problem is

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^n} -\psi^*(\lambda\boldsymbol{\alpha}) - \frac{\lambda}{2}\Omega^*(\boldsymbol{X}^\top\boldsymbol{\alpha})^2. \tag{1.19}$$

If Ω is the Euclidean norm (i.e., the ℓ_2-norm) then the previous problem is simply

$$G(\boldsymbol{K}) := \max_{\boldsymbol{\alpha}\in\mathbb{R}^n} -\psi^*(\lambda\boldsymbol{\alpha}) - \frac{\lambda}{2}\boldsymbol{\alpha}^\top\boldsymbol{K}\boldsymbol{\alpha} \quad \text{with} \quad \boldsymbol{K} = \boldsymbol{X}\boldsymbol{X}^\top. \tag{1.20}$$

Focusing on this last case, a few remarks are crucial:

1. The dual problem depends on the design $\boldsymbol{X}$ only through the kernel matrix $\boldsymbol{K} \in \mathbb{R}^{n\times n}$.
2. G is a *convex* function of $\boldsymbol{K}$ (as a maximum of linear functions).
3. The solutions $\boldsymbol{w}^*$ and $\boldsymbol{\alpha}^*$ to the primal and dual problems satisfy $\boldsymbol{w}^* = \boldsymbol{X}^\top\boldsymbol{\alpha}^* = \sum_{i=1}^n \boldsymbol{\alpha}_i^*\boldsymbol{x}_i$.
4. The exact same duality result applies for the generalization to $\boldsymbol{w}, \boldsymbol{x}_i \in \mathcal{H}$ for $\mathcal{H}$ a Hilbert space.

The first remark suggests a way to solve learning problems that are non-linear in the inputs $\boldsymbol{x}_i$: in particular consider a non-linear mapping ϕ which maps $\boldsymbol{x}_i$ to a high-dimensional $\phi(\boldsymbol{x}_i) \in \mathcal{H}$ with $\mathcal{H} = \mathbb{R}^d$ for $d \gg p$ or possibly an infinite dimensional Hilbert space. Then problem (1.15) with $f(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^n \ell(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle, y_i)$ becomes high-dimensional to solve in the primal, while it is simply solved in the dual by choosing a kernel matrix with entries $\boldsymbol{K}_{i,j} = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)\rangle$, which is advantageous as soon as $n^2 \leq d$; this is the so-called "kernel trick" (see more details in [105, 109]).

In particular if we consider functions $h \in \mathcal{H}$ where $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS) with reproducing kernel K then

$$\min_{h \in \mathcal{H}} \frac{1}{n}\sum_{i=1}^n \ell(h(\boldsymbol{x}_i), y_i) + \frac{\lambda}{2}\|h\|_{\mathcal{H}}^2 \tag{1.21}$$

is solved by solving Eq. (1.20) with $\boldsymbol{K}_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$. When applied to the mapping $\phi : \boldsymbol{x} \mapsto K(\boldsymbol{x}, \cdot)$, the third remark above yields the representer theorem of Kimmeldorf and Wahba [64][8] stating that $h^*(\cdot) = \sum_{i=1}^n \alpha_i^* K(\boldsymbol{x}_i, \cdot)$.

The fact that G is convex function of $\boldsymbol{K}$ suggests the possibility of optimizing the objective with respect to the choice of the kernel itself by solving a problem of the form $\min_{\boldsymbol{K} \in \mathcal{K}} G(\boldsymbol{K})$ where $\mathcal{K}$ is a convex set of kernel matrices.

In particular, given a finite set of kernel functions $(K_i)_{1 \leq i \leq p}$ it is natural to consider to find the best *linear* combination of kernels, which requires to add a positive definiteness constraint on the kernel, leading to a semi-definite program [68]:

$$\min_{\boldsymbol{\eta} \in \mathbb{R}^p} G(\textstyle\sum_{i=1}^p \boldsymbol{\eta}_i \boldsymbol{K}_i) \;\text{ s.t. }\; \sum_{i=1}^p \eta_i \boldsymbol{K}_i \succeq 0,\; \operatorname{tr}(\sum_{i=1}^p \boldsymbol{\eta}_i \boldsymbol{K}_i) \leq 1. \tag{1.22}$$

Assuming that the kernels have equal trace, the two constraints of the previous program are avoided by considering convex combinations of kernels, which leads to a quadratically-constrained quadratic program (QCQP) [67]:

$$\min_{\boldsymbol{\eta} \in \mathbb{R}_+^p} G(\textstyle\sum_{i=1}^p \boldsymbol{\eta}_i \boldsymbol{K}_i) \quad \text{s.t.} \quad \sum_{i=1}^p \boldsymbol{\eta}_i = 1. \tag{1.23}$$

We now present a reformulation of Eq. (1.23) using sparsity-inducing norms (see [7, 13, 96] for more details).

1.5.1 Variational formulation for sums of ℓ_2-norms

Variational formulations of structured norms are quite useful, among others to propose a general algorithmic scheme for learning problems regularized with this norm. We introduce them here because they provide the simplest link with the multiple kernel learning framework. See Section 5 for a generalization to all possible norms.

[8] Note that this provides a proof of the representer theorem for *convex* losses only and that the parameters $\boldsymbol{\alpha}$ are obtained through a dual *maximization* problem.

The two basic variational identity we use are

$$2ab = \inf_{\eta \in \mathbb{R}_+^*} \eta^{-1} a^2 + \eta\, b^2, \tag{1.24}$$

where the infimum is attained at $\eta = a/b$, and for $\boldsymbol{a} \in \mathbb{R}_+^p$,

$$\big(\sum_{i=1}^p \boldsymbol{a}_i\big)^2 = \inf_{\boldsymbol{\eta} \in (\mathbb{R}_+^*)^p} \sum_{i=1}^p \frac{\boldsymbol{a}_i^2}{\boldsymbol{\eta}_i} \;\text{ s.t. }\; \sum_{i=1}^p \boldsymbol{\eta}_i = 1. \tag{1.25}$$

The last inequality is a direct consequence of the Cauchy-Schwartz inequality:

$$\sum_{i=1}^p \boldsymbol{a}_i = \sum_{i=1}^p \frac{\boldsymbol{a}_i}{\sqrt{\boldsymbol{\eta}_i}} \cdot \sqrt{\boldsymbol{\eta}_i} \le \big(\sum_{i=1}^p \frac{\boldsymbol{a}_i^2}{\boldsymbol{\eta}_i}\big)^{1/2} \big(\sum_{i=1}^p \boldsymbol{\eta}_i\big)^{1/2}. \tag{1.26}$$

The infima in the previous expressions can be replaced by a minimization if the function $(x, y) \mapsto \frac{x}{y}$ is extended in $(0, 0)$ using the convention "0/0=0", since the resulting function is a proper closed convex function. We will use this convention implicitly from now on. Applying these variational forms to the ℓ_1- and ℓ_1/ℓ_2-norms (with non overlapping groups) with $\|\boldsymbol{w}\|_{\ell_1/\ell_2} = \sum_{g \in \mathcal{G}} \|\boldsymbol{w}_g\|_2$, with $|\mathcal{G}| = m$, and with $\triangle_p = \{\boldsymbol{\eta} \in \mathbb{R}_+^p \mid \sum_{i=1}^p \boldsymbol{\eta}_i = 1\}$ the simplex, we obtain directly:

$$\|\boldsymbol{w}\|_1 = \min_{\boldsymbol{\eta} \in \mathbb{R}_+^p} \frac{1}{2} \sum_{i=1}^p \big[\frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i} + \boldsymbol{\eta}_i\big], \qquad \|\boldsymbol{w}\|_1^2 = \min_{\boldsymbol{\eta} \in \triangle_p} \sum_{i=1}^p \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i},$$

$$\|\boldsymbol{w}\|_{\ell_1/\ell_2} = \min_{\boldsymbol{\eta} \in \mathbb{R}_+^m} \frac{1}{2} \sum_{g \in \mathcal{G}} \big[\frac{\|\boldsymbol{w}_g\|^2}{\boldsymbol{\eta}^g} + \boldsymbol{\eta}^g\big], \qquad \|\boldsymbol{w}\|_{\ell_1/\ell_2}^2 = \min_{\boldsymbol{\eta} \in \triangle_m} \sum_{g \in \mathcal{G}} \frac{\|\boldsymbol{w}_g\|^2}{\boldsymbol{\eta}^g}.$$

These formulations have appeared in [60, 95, 96] and many extensions exist in the literature. See further extensions in Sections 1.5.3 and 5.

1.5.2 From the group Lasso to MKL

With the previous variational formulation in hand the connection between the group Lasso and MKL is almost immediate. We indeed have, assuming that $\mathcal{G}$ is a partition of $\{1, \dots, p\}$,

$$\begin{aligned}
&\min_{\boldsymbol{w} \in \mathbb{R}^p} && \psi(\boldsymbol{X}\boldsymbol{w}) + \frac{\lambda}{2}\big(\textstyle\sum_{g \in \mathcal{G}} \|\boldsymbol{w}_g\|_2\big)^2 \\
&= \min_{\boldsymbol{w} \in \mathbb{R}^p, \boldsymbol{\eta} \in \triangle_m} && \psi(\boldsymbol{X}\boldsymbol{w}) + \frac{\lambda}{2} \sum_{g \in \mathcal{G}} \frac{\|\boldsymbol{w}_g\|_2^2}{\boldsymbol{\eta}_g} \\
&= \min_{\widetilde{\boldsymbol{w}} \in \mathbb{R}^p, \boldsymbol{\eta} \in \triangle_m} && \psi(\textstyle\sum_{g \in \mathcal{G}} \boldsymbol{\eta}_g^{1/2} \boldsymbol{X}_g \widetilde{\boldsymbol{w}}_g) + \frac{\lambda}{2} \sum_{g \in \mathcal{G}} \|\widetilde{\boldsymbol{w}}_g\|_2^2 \\
&= \min_{\widetilde{\boldsymbol{w}} \in \mathbb{R}^p, \boldsymbol{\eta} \in \triangle_m} && \psi(\widetilde{\boldsymbol{X}}\widetilde{\boldsymbol{w}}) + \frac{\lambda}{2}\|\widetilde{\boldsymbol{w}}\|_2^2 \quad \text{s.t. } \widetilde{\boldsymbol{X}} = [\eta_{g_1}^{1/2} \boldsymbol{X}_{g_1}, \dots, \eta_{g_m}^{1/2} \boldsymbol{X}_{g_m}] \\
&= \min_{\boldsymbol{\eta} \in \triangle_m} \max_{\boldsymbol{\alpha} \in \mathbb{R}^n} && -\psi^*(\lambda\boldsymbol{\alpha}) - \frac{\lambda}{2}\boldsymbol{\alpha}^\top \big(\textstyle\sum_{g \in \mathcal{G}} \boldsymbol{\eta}_g \boldsymbol{K}_g\big)\boldsymbol{\alpha} \\
&= \min_{\boldsymbol{\eta} \in \triangle_m} && G(\textstyle\sum_{g \in \mathcal{G}} \boldsymbol{\eta}_g \boldsymbol{K}_g),
\end{aligned}$$

where the third line results from the change of variable $\widetilde{\boldsymbol{w}}_g \boldsymbol{\eta}_g = \boldsymbol{w}_g$.

Note that the Lasso corresponds to the special case where groups are singletons and where $\boldsymbol{K}_i = \boldsymbol{x}_i \boldsymbol{x}_i^\top$ is a rank-one kernel matrix. In other words, MKL with rank-one kernel matrices (i.e., feature spaces of dimension one) is equivalent to ℓ_1-regularization and thus simpler algorithms can be brought to bear in this situation.

We have shown that learning convex combinations of kernels through Eq. (1.23) turns out to be equivalent to an ℓ_1/ℓ_2-norm penalized problems. In other words, learning a linear combination $\sum_{i=1}^m \boldsymbol{\eta}_i \boldsymbol{K}_i$ of kernel matrices, subject to $\boldsymbol{\eta}$ in the simplex $\triangle_m$ is equivalent to penalizing the empirical risk with an ℓ_1-norm applied to norms of predictors $\|\boldsymbol{w}_g\|_2$. This link between the ℓ_1-norm and the simplex may be extended to other norms, namely "subquadratic" norms, which are associated to other compact sets, and which we now describe.

1.5.3 Variational formulations for subquadratic norms

We have seen that for $\boldsymbol{w} \in \mathbb{R}^p$, $\|\boldsymbol{w}\|_1^2 = \min_{\boldsymbol{\eta} \in \triangle_p} \sum_{i=1}^p \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i}$. We now show that similar representations are possible for certain norms.

Lemma 1.1 *Let Ω and Ω^* a pair of dual norms. Define $\bar{\Omega}^*(\boldsymbol{w}) = \Omega(\boldsymbol{w}^{1/2})^2$ where we noted $\boldsymbol{w}^{1/2} = (\boldsymbol{w}_1^{1/2}, \dots, \boldsymbol{w}_p^{1/2})^\top$. If $\bar{\Omega}^*$ is a convex function — we then say that Ω is* subquadratic*, it is a norm and if $\bar{\Omega}$ denotes the corresponding primal norm then*

$$\begin{aligned}
\Omega(\boldsymbol{w}) &= \frac{1}{2} \min_{\boldsymbol{\eta} \in \mathbb{R}_+^p} \sum_i \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i} + \bar{\Omega}(\boldsymbol{\eta}) \\
\Omega(\boldsymbol{w})^2 &= \min_{\boldsymbol{\eta} \in H} \sum_i \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i} \text{ where } H = \{\boldsymbol{\eta},\ \bar{\Omega}(\boldsymbol{\eta}) \leq 1\}.
\end{aligned}$$

Proof Note that by construction, $\bar{\Omega}^*$ is homogenous, symmetric and definite ($\bar{\Omega}^*(\boldsymbol{\kappa}) = 0 \Rightarrow \boldsymbol{\kappa} = 0$). If $\bar{\Omega}^*$ is convex then $\bar{\Omega}^*(\frac{1}{2}(\boldsymbol{v}+\boldsymbol{u})) \leq \frac{1}{2}\big(\bar{\Omega}^*(\boldsymbol{v}) + \bar{\Omega}^*(\boldsymbol{u})\big)$, which by homogeneity shows that $\bar{\Omega}^*$ also satisfies the triangle inequality. Together, these properties show that $\bar{\Omega}^*$ is a norm. For the first identity we have

$$\begin{aligned}
\Omega(\boldsymbol{w}) &= \max_{\boldsymbol{\kappa} \in \mathbb{R}^p} \boldsymbol{\kappa}^\top |\boldsymbol{w}| \quad \text{s.t.} \quad \Omega^*(\boldsymbol{\kappa}) \leq 1 \\
&= \max_{\boldsymbol{\kappa} \in \mathbb{R}_+^p} \sum_{i=1}^p \boldsymbol{\kappa}_i^{1/2} |\boldsymbol{w}_i| \quad \text{s.t.} \quad \Omega^*(\boldsymbol{\kappa}^{1/2})^2 \leq 1 \\
&= \max_{\boldsymbol{\kappa} \in \mathbb{R}_+^p} \min_{\boldsymbol{\eta} \in \mathbb{R}_+^p} \frac{1}{2} \sum_{i=1}^p \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i} + \boldsymbol{\kappa}^\top \boldsymbol{\eta} \quad \text{s.t.} \quad \bar{\Omega}^*(\boldsymbol{\kappa}) \leq 1 \\
&= \min_{\boldsymbol{\eta} \in \mathbb{R}_+^p} \max_{\boldsymbol{\kappa} \in \mathbb{R}_+^p} \frac{1}{2} \sum_{i=1}^p \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i} + \boldsymbol{\kappa}^\top \boldsymbol{\eta} \quad \text{s.t.} \quad \bar{\Omega}^*(\boldsymbol{\kappa}) \leq 1,
\end{aligned}$$

which proves the first variational formulation. The second one follows similarly by applying (1.25) instead of (1.24).

$$\begin{aligned}
\Omega(\boldsymbol{w})^2 &= \max_{\boldsymbol{\kappa}\in\mathbb{R}^p} \big(\sum_{i=1}^p \boldsymbol{\kappa}_i^{1/2}|\boldsymbol{w}_i|\big)^2 \quad \text{s.t.} \quad \Omega^*(\boldsymbol{\kappa}^{1/2})^2 \leq 1 \\
&= \max_{\boldsymbol{\kappa}\in\mathbb{R}^p_+} \min_{\tilde{\boldsymbol{\eta}}\in\mathbb{R}^p_+} \frac{1}{2}\sum_{i=1}^p \frac{\boldsymbol{\kappa}_i \boldsymbol{w}_i^2}{\tilde{\boldsymbol{\eta}}_i} \quad \text{s.t.} \quad \sum_i \tilde{\boldsymbol{\eta}}_i = 1,\ \bar{\Omega}^*(\boldsymbol{\kappa}) \leq 1 \\
&= \max_{\boldsymbol{\kappa}\in\mathbb{R}^p_+} \min_{\boldsymbol{\eta}\in\mathbb{R}^p_+} \frac{1}{2}\sum_{i=1}^p \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i} \quad \text{s.t.} \quad \boldsymbol{\eta}^\top\boldsymbol{\kappa} = 1,\ \bar{\Omega}^*(\boldsymbol{\kappa}) \leq 1
\end{aligned}$$

■

Thus, given a sub-quadratic norm, we may define a convex set H, namely the unit ball of $\bar{\Omega}$, such that $\Omega(\boldsymbol{w})^2 = \min_{\boldsymbol{\eta}\in H} \sum_{i=1}^p \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i}$. Norms defined by this last variational formulation for a given convex set H have been recently studied in [81].

We show that, for such norms, the dual norm has an explicit form:

Lemma 1.2 *Define $\Omega_H(\boldsymbol{w})^2 = \min_{\boldsymbol{\eta}\in H} \sum_{i=1}^p \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i}$. Then, if H is a compact convex set, Ω_H is a norm and the dual norm satisfies $\Omega_H^*(\boldsymbol{\kappa})^2 = \max_{\boldsymbol{\eta}\in H} \sum_{i=1}^p \boldsymbol{\eta}_i \boldsymbol{\kappa}_i^2$.*

Proof Symmetry, positive definiteness and homogeneity of Ω are straightforward from the definitions. Ω is convex, since it is obtained by minimization of $\boldsymbol{\eta}$ in a jointly convex formulation. Thus Ω is a norm. Finally

$$\begin{aligned}
\frac{1}{2}\Omega_H^*(\boldsymbol{\kappa})^2 &= \max_{\boldsymbol{w}\in\mathbb{R}^p} \boldsymbol{w}^\top\boldsymbol{\kappa} - \frac{1}{2}\Omega_H(\boldsymbol{w})^2 \\
&= \max_{\boldsymbol{w}\in\mathbb{R}^p} \max_{\boldsymbol{\eta}\in H} \boldsymbol{w}^\top\boldsymbol{\kappa} - \frac{1}{2}\boldsymbol{w}^\top \mathrm{Diag}(\boldsymbol{\eta})^{-1}\boldsymbol{w}.
\end{aligned}$$

The form of the dual norm follows by maximizing out w.r.t. $\boldsymbol{w}$. ■

For norms that are not subquadratic, it is often the case that their dual norm is itself subquadratic is which case symmetric variational forms can be obtained [2]. Finally, we show in section 5 that all norms admit a quadratic variational form provided the bilinear form considered is allowed to be non-diagonal.

1.5.4 Structured multiple kernel learning

For norms that have a variational form as in Lemma 1.2, we can generalize the equivalence of the regularization by an ℓ_1/ℓ_2-norm with MKL to other structured norms. We have:

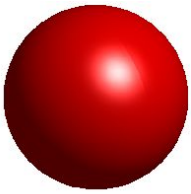
$$
\begin{aligned}
&\min_{\boldsymbol{w}\in\mathbb{R}^p} && \psi(\boldsymbol{X}\boldsymbol{w}) + \frac{\lambda}{2}\Omega_H(\boldsymbol{w})^2 \\
=&\min_{\boldsymbol{w}\in\mathbb{R}^p,\boldsymbol{\eta}\in H} && \psi(\boldsymbol{X}\boldsymbol{w}) + \frac{\lambda}{2}\sum_{i=1}^p \frac{\boldsymbol{w}_i^2}{\boldsymbol{\eta}_i} \\
=&\min_{\widetilde{\boldsymbol{w}}\in\mathbb{R}^p,\boldsymbol{\eta}\in H} && \psi(\textstyle\sum_{i=1}^p \boldsymbol{\eta}_i^{1/2}\boldsymbol{X}_i\widetilde{\boldsymbol{w}}_i) + \frac{\lambda}{2}\sum_{i=1}^p \widetilde{\boldsymbol{w}}_i^2 \\
=&\min_{\widetilde{\boldsymbol{w}}\in\mathbb{R}^p,\boldsymbol{\eta}\in H} && \psi(\widetilde{\boldsymbol{X}}\widetilde{\boldsymbol{w}}) + \frac{\lambda}{2}\|\widetilde{\boldsymbol{w}}\|_2^2 \quad \text{s.t. } \widetilde{\boldsymbol{X}} = [\boldsymbol{\eta}_1^{1/2}\boldsymbol{X}_1,\dots,\boldsymbol{\eta}_p^{1/2}\boldsymbol{X}_p] \\
=&\min_{\boldsymbol{\eta}\in H}\max_{\boldsymbol{\alpha}\in\mathbb{R}^n} && -\psi^*(\lambda\boldsymbol{\alpha}) - \frac{\lambda}{2}\boldsymbol{\alpha}^\top\big(\textstyle\sum_{i=1}^p \boldsymbol{\eta}_i\boldsymbol{K}_i\big)\boldsymbol{\alpha} \\
=&\min_{\boldsymbol{\eta}\in H} && G(\textstyle\sum_{i=1}^p \boldsymbol{\eta}_i\boldsymbol{K}_i).
\end{aligned}
\tag{1.27}
$$

This suggests a natural extension to the RKHS settings of structured norms. Indeed let, $h = (h_1,\dots,h_p) \in \mathcal{B} := \mathcal{H}_1 \times \dots \times \mathcal{H}_p$, where $\mathcal{H}_i$ are RKHSes. It is easy to verify that $h \mapsto \Omega_H\big((\|h_1\|_{\mathcal{H}_1},\dots,\|h_p\|_{\mathcal{H}_p}\big)$ is a norm, using the variational formulation of Ω_H. Moreover, the learning problem

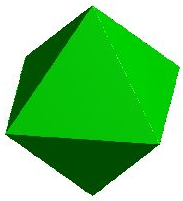
$$
\min_{h\in\mathcal{B}} \psi(h_1(\boldsymbol{x}) + \dots + h_p(\boldsymbol{x})) + \frac{\lambda}{2}\Omega_H\big((\|h_1\|_{\mathcal{H}_1},\dots,\|h_p\|_{\mathcal{H}_p})^2\big) \tag{1.28}
$$

is equivalent, by the above derivation, to the MKL problem $\min_{\boldsymbol{\eta}\in H} G(\sum_{i=1}^p \boldsymbol{\eta}_i\boldsymbol{K}_i)$ with $[\boldsymbol{K}_i]_{j,j'} = K_i(\boldsymbol{x}_j,\boldsymbol{x}_{j'})$ for K_i the reproducing kernel of $\mathcal{H}_i$.

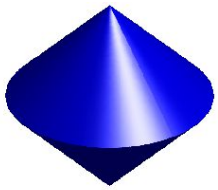
Thus, for most of the structured sparsity-inducing norms that we have considered in Section 1.3, we may replace individual variables by whole Hilbert spaces. For example, tree-structured sparsity (and its extension to directed acyclic graphs) was explored in [9] where each node of the graph was a RKHS, with application to non-linear variable selection.



(a) ℓ_2-norm ball



(b) ℓ_1-norm ball

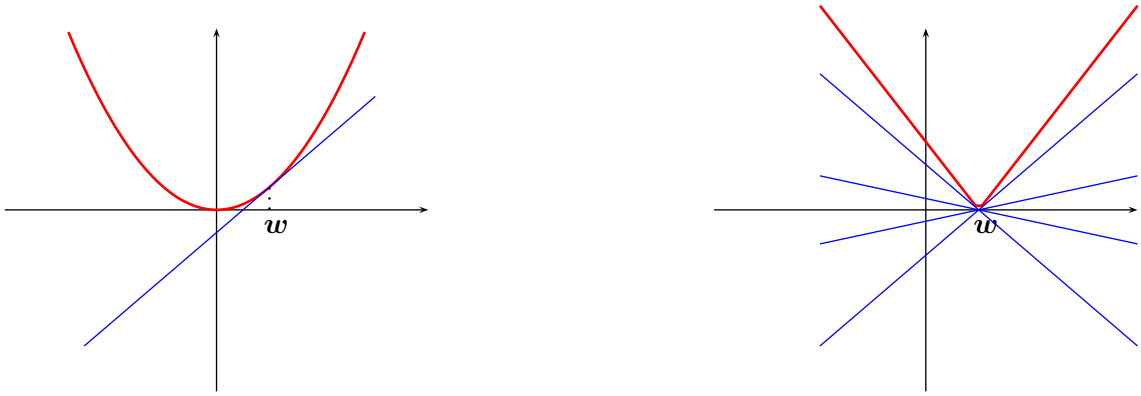


(c) ℓ_1/ℓ_2-norm ball without overlaps: $\Omega(\boldsymbol{w}) = \|\boldsymbol{w}_{\{1,2\}}\|_2 + |\boldsymbol{w}_3|$



(d) ℓ_1/ℓ_2-norm ball with overlaps: $\Omega(\boldsymbol{w}) = \|\boldsymbol{w}_{\{1,2,3\}}\|_2 + |\boldsymbol{w}_1| + |\boldsymbol{w}_2|$

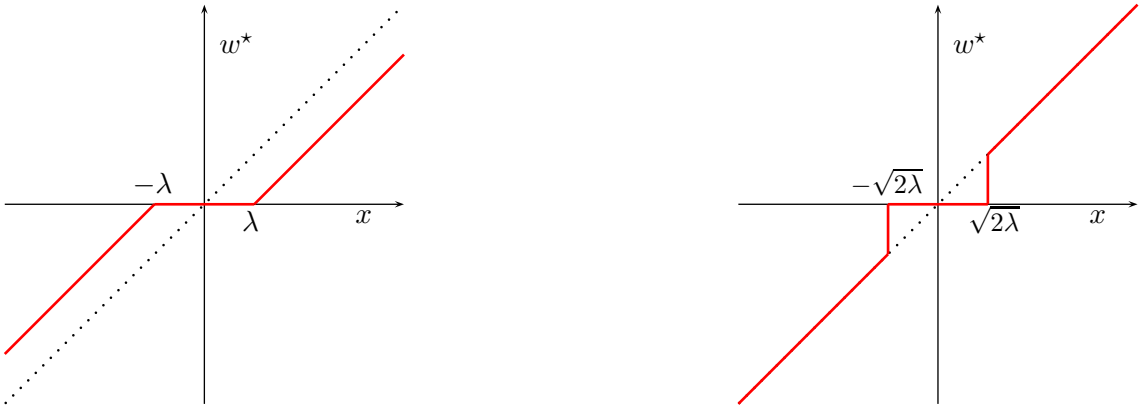
Figure 1.4: Comparison between different balls of sparsity-inducing norms in three dimensions. The singular points appearing on these balls describe the sparsity-inducing behavior of the underlying norms Ω.



(a) Smooth case

(b) Non-smooth case

Figure 1.5: Gradients and subgradients for smooth and non-smooth functions.



(a) soft-thresholding operator,
$w^\star = \mathrm{sign}(x)(|x| - \lambda)_+$,
$\min_w \frac{1}{2}(x-w)^2 + \lambda|w|$

(b) hard-thresholding operator
$w^\star = \mathbf{1}_{|x| \geq \sqrt{2\lambda}} x$
$\min_w \frac{1}{2}(x-w)^2 + \lambda \mathbf{1}_{|w|>0}$

Figure 1.6: Soft- and hard-thresholding operators.

Chapter 2

Generic Methods

The problem defined in Eq. (1.1) is convex, as soon as both the loss f and the regularizer Ω are convex functions. In this section, we consider optimization strategies which are essentially blind to problem structure, namely subgradient descent (see, e.g., [18]), which is applicable under weak assumptions, and interior point methods solving reformulations such as linear programs (LP), quadratic programs (QP) or more generally, second-order cone programming (SOCP) or semidefinite programming (SDP) problems (see, e.g., [25]). The latter strategy is usually only possible with the square loss and makes use of general-purpose optimization toolboxes.

Subgradient descent. For all convex unconstrained problems, subgradient descent can be used as soon as one subgradient can be computed efficiently. In our setting, this is possible when a subgradient of the loss f, and a subgradient of the regularizer Ω can be computed. This is true for all the norms that we have considered, and leads to the following iterative algorithm

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{\alpha}{t^\beta}(\boldsymbol{s} + \lambda \boldsymbol{s}'), \text{ where } \boldsymbol{s} \in \partial f(\boldsymbol{w}_t), \ \boldsymbol{s}' \in \partial \Omega(\boldsymbol{w}_t)$$

with α a well-chosen positive parameter and β is typically 1 or 1/2. Under certain conditions, these updates are globally convergent. More precisely, we have, from [87], $F(\boldsymbol{w}_t) - \min_{\boldsymbol{w} \in \mathbb{R}^p} F(\boldsymbol{w}) = O(\frac{\log t}{\sqrt{t}})$ for Lipschitz-continuous function and $\beta = 1/2$. However, the convergence is in practice slow (i.e., many iterations are needed), and the solutions obtained are usually not sparse. This is to be contrasted with the proximal methods presented in the next section which are less generic but more adapted to sparse problems.

Reformulation as LP, QP, SOCP, SDP. For all the sparsity-inducing norms we consider in this chapter the corresponding regularized least-square problem can be represented by standard mathematical programming problems, all of them being SDPs, and often simpler (e.g., QP). For example, for the ℓ_1-norm regularized least-square regression, we can

reformulate $\min_{w \in \mathbb{R}^p} \frac{1}{2n} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda \Omega(\boldsymbol{w})$ as

$$\min_{\boldsymbol{w}_+, \boldsymbol{w}_- \in \mathbb{R}_+^p} \frac{1}{2n} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}_+ + \boldsymbol{X}\boldsymbol{w}_-\|_2^2 + \lambda (1^\top \boldsymbol{w}_+ + 1^\top \boldsymbol{w}_-),$$

which is a quadratic program. Other problems can be similarly cast (for the trace norm, see [8, 44]).

General-purpose toolboxes can then be used, to get solutions with high precision (low duality gap). However, in the context of machine learning, this is inefficient for two reasons: (1) these toolboxes are generic and blind to problem structure and tend to be too slow, or cannot even run because of memory problems, (2) as outlined by [22], high precision is not necessary for machine learning problems, and a duality gap of the order of machine precision (which would be a typical result from toolboxes) is not necessary.

We now present in the following sections methods that are adapted to problems regularized by sparsity-inducing norms.

Chapter 3

Proximal Methods

3.1 Principle of Proximal Methods

Proximal methods are specifically tailored to optimize an objective of the form (1.1), i.e., which can be written as the sum of a generic smooth differentiable function f with Lipschitz-continuous gradient, and a non-differentiable function $\lambda\Omega$. They have drawn increasing attention in the machine learning community, especially because of their convergence rates (optimal for the class of first-order techniques) and their ability to deal with large nonsmooth convex problems (e.g., [17, 34, 89, 127]).

Proximal methods can be described as follows: at each iteration the function f is linearized around the current point and a problem of the form

$$\min_{\boldsymbol{w}\in\mathbb{R}^p} f(\boldsymbol{w}^t)+\nabla f(\boldsymbol{w}^t)^T(\boldsymbol{w}-\boldsymbol{w}^t)+\lambda\Omega(\boldsymbol{w})+\frac{L}{2}\|\boldsymbol{w}-\boldsymbol{w}^t\|_2^2 \tag{3.1}$$

is solved. The quadratic term, called proximal term, keeps the update in a neighborhood of the current iterate $\boldsymbol{w}^t$ where f is close to its linear approximation; $L>0$ is a parameter, which should essentially be an upper bound on the Lipschitz constant of ∇f and is typically set with a linesearch. This problem can be rewritten as

$$\min_{\boldsymbol{w}\in\mathbb{R}^p} \frac{1}{2}\|\boldsymbol{w}-\big(\,\boldsymbol{w}^t-\tfrac{1}{L}\nabla f(\boldsymbol{w}^t)\,\big)\|_2^2+\tfrac{\lambda}{L}\Omega(\boldsymbol{w}). \tag{3.2}$$

It should be noted that when the nonsmooth term Ω is not present, the solution of the previous proximal problem just yields the standard gradient update rule $\boldsymbol{w}^{t+1}\leftarrow\boldsymbol{w}^t-\frac{1}{L}\nabla f(\boldsymbol{w}^t)$. Furthermore, if Ω is the indicator function of a set ι_C, i.e., defined by $\iota_C(x)=0$ for $x\in C$ and $\iota_C(x)=+\infty$ otherwise, then solving (3.2) yields the projected gradient update with projection on the set C. This suggests that the solution of the proximal problem provides an interesting generalization of gradient updates, and motivates the introduction of the notion of a *proximal operator* associated with the regularization term $\lambda\Omega$.

The proximal operator, which we will denote $\text{Prox}_{\mu\Omega}$, was defined by [82] as the function that maps a vector $\boldsymbol{u} \in \mathbb{R}^p$ to the unique[1] solution of

$$\min_{\boldsymbol{w}\in\mathbb{R}^p} \frac{1}{2}\|\boldsymbol{u}-\boldsymbol{w}\|_2^2 + \mu\,\Omega(\boldsymbol{w}). \tag{3.3}$$

This operator is clearly central to proximal methods since their main step consists in computing $\text{Prox}_{\frac{\lambda}{L}\Omega}\big(\boldsymbol{w}^t - \frac{1}{L}\nabla f(\boldsymbol{w}^t)\big)$.

In section 3.3, we present analytical forms of proximal operators associated with simple norms and algorithms to compute them in some more elaborate cases. Note that the proximal term in Eq. (3.1) could be replaced by any Bregman divergences (see, e.g., [118]), which may be useful in settings where extra constraint (such as non-negativity) are added to the problem.

3.2 Algorithms

The basic proximal algorithm uses the solution of problem (3.2) as the next update $\boldsymbol{w}^{t+1}$; however fast variants such as the accelerated algorithm presented in [89] or FISTA [17] maintain two variables and use them to combine at marginal extra computational cost the solution of (3.2) with information about previous steps. Often, an upper bound on the Lipschitz constant of ∇f is not known, and even if it is[2], it is often better to obtain a local estimate. A suitable value for L can be obtained by iteratively increasing L by a constant factor until the condition

$$f(\boldsymbol{w}_L^\star) \leq f(\boldsymbol{w}^t) + \nabla f(\boldsymbol{w}^t)^T(\boldsymbol{w}_L^\star - \boldsymbol{w}^t) + \tfrac{L}{2}\|\boldsymbol{w}_L^\star - \boldsymbol{w}^t\|_2^2 \tag{3.4}$$

is met, where $\boldsymbol{w}_L^\star$ denotes the solution of (3.2).

For functions f whose gradients are Lipschitz-continuous, the basic proximal algorithm has a global convergence rate in $O(\frac{1}{t})$ where t is the number of iterations of the algorithm. Accelerated algorithms like FISTA can be shown to have global convergence rate in $O(\frac{1}{t^2})$. Perhaps more importantly, both basic (ISTA) and accelerated [89] proximal methods are adaptive in the sense that if f is strongly convex — and the problem is therefore better conditioned — the convergence is actually linear (i.e., with rates in $O(C^t)$ for some constant $C < 1$; see [89]). Finally, it should be noted that accelerated schemes are not necessarily descent algorithms, in the sense that the objective does not necessarily decrease at each iteration in spite of the global convergence properties.

[1] Since the objective is strongly convex.

[2] For problems common in machine learning where $f(\boldsymbol{w}) = \psi(\boldsymbol{X}\boldsymbol{w})$ and ψ is twice differentiable, then L may be chosen to be the largest eigenvalue of $\frac{1}{n}\boldsymbol{X}^T\boldsymbol{X}$ times the supremum over $\boldsymbol{u} \in \mathbb{R}^n$ of the largest eigenvalue of the Hessian of ψ at $\boldsymbol{u}$.

3.3 Computing the Proximal Operator

Computing the *proximal operator efficiently* and *exactly* is crucial to enjoy the fast convergence rates of proximal methods. We therefore focus here on properties of this operator and on its computation for several sparsity-inducing norms.

Dual proximal operator. In the case where Ω is a norm, by Fenchel duality, the following problem is dual (see Proposition 1.3) to problem (3.2):

$$\max_{\boldsymbol{v}\in\mathbb{R}^p} -\frac{1}{2}\left[\|\boldsymbol{v}-\boldsymbol{u}\|_2^2 - \|\boldsymbol{u}\|^2\right] \quad \text{such that} \quad \Omega^*(\boldsymbol{v}) \leq \mu. \tag{3.5}$$

Lemma 3.1 (Relation to dual proximal operator) *Let $Prox_{\mu\Omega}$ be the proximal operator associated with the regularization $\mu\Omega$, where Ω is a norm, and $Proj_{\{\Omega^*(\cdot)\leq\mu\}}$ be the projector on the ball of radius μ of the dual norm Ω^*. Then $Proj_{\{\Omega^*(\cdot)\leq\mu\}}$ is the proximal operator for the dual problem (3.5) and, denoting the identity I_d, these two operators satisfy the relation*

$$Prox_{\mu\Omega} = I_d - Proj_{\{\Omega^*(\cdot)\leq\mu\}}. \tag{3.6}$$

Proof *By Proposition 1.3, if $\boldsymbol{w}^\star$ is optimal for (3.3) and $\boldsymbol{v}^\star$ is optimal for (3.5), we have*[3] *$-\boldsymbol{v}^\star = \nabla f(\boldsymbol{w}^\star) = \boldsymbol{w}^\star - \boldsymbol{u}$. Since $\boldsymbol{v}^\star$ is the projection of $\boldsymbol{u}$ on the ball of radius μ of the norm Ω^*, the result follows.*

This lemma shows that the proximal operator can always be computed as the residual of a projection onto a convex set. More general results appear in [35].

ℓ_1-norm regularization. Using optimality conditions for (3.5) and then (3.6) or subgradient condition (1.7) applied to (3.3), it is easy to check that $\text{Proj}_{\{\|\cdot\|_\infty\leq\mu\}}$ and $\text{Prox}_{\mu\|\cdot\|_1}$ respectively satisfy:

$$\left[\text{Proj}_{\{\|\cdot\|_\infty\leq\mu\}}(\boldsymbol{u})\right]_j = \min\left(1, \tfrac{\mu}{|\boldsymbol{u}_j|}\right)\boldsymbol{u}_j,$$

and

$$\left[\text{Prox}_{\mu\|\cdot\|_1}(\boldsymbol{u})\right]_j = \left(1 - \frac{\mu}{|\boldsymbol{u}_j|}\right)_+ \boldsymbol{u}_j,$$

for $j \in \{1,\ldots,p\}$, with $(x)_+ := \max(x,0)$. Note that $\text{Prox}_{\mu\|\cdot\|_1}$ is componentwise the *soft-thresholding operator* of [38] presented in Section 1.4.

ℓ_1-norm constraint. Sometimes, the ℓ_1-norm is used as a hard constraint and, in that case, the optimization problem is

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}) \quad \text{such that} \quad \|\boldsymbol{w}\|_1 \leq C.$$

[3] The dual variable from Fenchel duality is $-\boldsymbol{v}$ in this case.

This problem can still be viewed as an instance of (1.1), with Ω defined by $\Omega(\boldsymbol{u}) = 0$ if $\|\boldsymbol{u}\|_1 \leq C$ and $\Omega(\boldsymbol{u}) = +\infty$ otherwise. Proximal methods thus apply and the corresponding proximal operator is the projection on the ℓ_1-ball, itself an instance of a *quadratic continuous knapsack problem* for which efficient pivot algorithms with linear complexity have been proposed [27, 73].

ℓ_1/ℓ_q-norm ("group Lasso"). If $\mathcal{G}$ is a partition of $\{1, \ldots, p\}$, the dual norm of the ℓ_1/ℓ_q norm is the $\ell_\infty/\ell_{q'}$ norm, with $\frac{1}{q} + \frac{1}{q'} = 1$. It is easy to show that the orthogonal projection on a unit $\ell_\infty/\ell_{q'}$ ball is obtained by projecting separately each subvector $\boldsymbol{u}_g$ on a unit $\ell_{q'}$-ball in $\mathbb{R}^{|g|}$. For the ℓ_1/ℓ_2-norm $\Omega : \boldsymbol{w} \mapsto \sum_{g\in\mathcal{G}} \|\boldsymbol{w}_g\|_2$ we have

$$[\mathrm{Prox}_{\mu\Omega}(\boldsymbol{u})]_g = \Big(1 - \frac{\lambda}{\|\boldsymbol{u}_g\|_2}\Big)_+ \boldsymbol{u}_g, \quad g \in \mathcal{G}.$$

This is shown easily by considering that the subgradient of the ℓ_2-norm is $\partial\|\boldsymbol{w}\|_2 = \big\{\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2}\big\}$ if $\boldsymbol{w} \neq \boldsymbol{0}$ or $\partial\|\boldsymbol{w}\|_2 = \{\boldsymbol{z} \mid \|\boldsymbol{z}\|_2 \leq 1\}$ if $\boldsymbol{w} = 0$ and by applying the result of Eq. (1.7).

For the ℓ_1/ℓ_∞-norm, whose dual norm is the ℓ_∞/ℓ_1-norm, an efficient algorithm to compute the proximal operator is based on Eq. (3.6). Indeed this equation indicates that the proximal operator can be computed on each group g as the residual of a projection on an ℓ_1-norm ball in $\mathbb{R}^{|g|}$; the latter is done efficiently with the previously mentioned linear-time algorithms.

In general, the case where groups overlap is more complicated because the regularization is no longer separable. Nonetheless, in some cases it is still possible to compute efficiently the proximal operator.

Hierarchical ℓ_1/ℓ_q-norms. Hierarchical norms were proposed by [132]. Following [59], we focus on the case of a norm $\Omega : \boldsymbol{w} \mapsto \sum_{g\in\mathcal{G}} \|\boldsymbol{w}_g\|_q$, with $q \in \{2, \infty\}$, where the set of groups $\mathcal{G}$ is *tree-structured*, meaning that two groups are either disjoint or one is included in the other. Let $\preceq$ be a total order such that $g_1 \preceq g_2$ if and only if either $g_1 \subset g_2$ or $g_1 \cap g_2 = \emptyset$.[4] Then, if $g_1 \preceq \ldots \preceq g_m$ with $m = |\mathcal{G}|$, and if we define Π_g as (a) the proximal operator $\boldsymbol{w}_g \mapsto \mathrm{Prox}_{\mu\|\cdot\|_q}(\boldsymbol{w}_g)$ on the subspace corresponding to group g and (b) the identity on the orthogonal, it can be shown [59] that:

$$\mathrm{Prox}_{\mu\Omega} = \Pi_{g_m} \circ \ldots \circ \Pi_{g_1}. \tag{3.7}$$

In other words, the proximal operator associated with the norm can be obtained as the composition of the proximal operators associated to individual groups provided that the ordering of the groups is well chosen. Note that this result does not hold for $q \notin \{1, 2, \infty\}$ (see [59] for more details).

Combined $\ell_1 + \ell_1/\ell_q$-norm ("sparse group Lasso"). The possibility of combining an ℓ_1/ℓ_q-norm that takes advantage of sparsity at the group level with an ℓ_1-norm that

[4] For a tree-structured $\mathcal{G}$ such an order exists.

induces sparsity within the groups is quite natural [45, 111]. Such regularizations are in fact a special case of the hierarchical ℓ_1/ℓ_q-norms presented above and the corresponding proximal operator is therefore readily computed by applying first soft-thresholding and then group soft-thresholding.

Overlapping ℓ_1/ℓ_∞-norms. When the groups overlap but do not have a tree structure, computing the proximal operator has proven to be more difficult, but it can still be done efficiently when $q = \infty$. Indeed, as shown by [76], there exists a dual relation between such an operator and a quadratic min-cost flow problem on a particular graph, which can be tackled using network flow optimization techniques. Moreover, it may be extended to more general situations where structured sparsity is expressed through submodular functions [10].

Trace norm. The proximal operator for the trace norm, i.e., the unique minimizer of $\frac{1}{2}\|\mathbf{M}-\mathbf{N}\|_F^2 + \lambda\|\mathbf{M}\|_*$ may obtained by computing a singular value decomposition of $\mathbf{N}$ and then replacing the singular values by their soft-thresholded versions.

3.4 Proximal methods for structured MKL

In this section we show how proximal methods can be applied to solve multiple kernel learning problems. More precisely, we follow [83] who showed, in the context of plain MKL that proximal algorithms are applicable in a RKHS. We extend and present here this idea to the general case of structured MKL, showing that the proximal operator for the structured RKHS norm may be obtained from the proximal operator of the corresponding subquadratic norms.

Given a collection of reproducing kernel Hilbert spaces $\mathcal{H}_1, \ldots, \mathcal{H}_p$, we consider the Cartesian product $\mathcal{B} := \mathcal{H}_1 \times \ldots \times \mathcal{H}_p$, equipped with the norm $\|h\|_\mathcal{B} := (\|h_1\|_{\mathcal{H}_1}^2 + \ldots + \|h_p\|_{\mathcal{H}_p}^2)^{1/2}$, where $h = (h_1, \ldots, h_p)$ with $h_i \in \mathcal{H}_i$.

The set $\mathcal{B}$ is a Hilbert space, in which gradients and subgradients are well defined and in which we can extend some algorithms that we considered in the Euclidean case easily.

In the following, we say that a norm is *monotonic* if $\Omega(\boldsymbol{w} + \gamma\, \boldsymbol{e}_i) > \Omega(\boldsymbol{w})$ when $\gamma\, \boldsymbol{w}_i > 0$. It is well-known that monotonic norms are exactly norms so that $\Omega(\boldsymbol{w})$ depends only of the absolute values $|\boldsymbol{w}_i|$ of the components $\boldsymbol{w}_i$, norms which are often referred to as *absolute* norms [16].

It should be noted that in particular all norms that have a diagonal quadratic variational form (see Lemma 1.2) are *monotonic*.

Lemma 3.2 *Let Ω be a* monotonic *norm on $\mathbb{R}^p$ with dual norm Ω^*, then $\Lambda : h \mapsto \Omega\big((\|h_1\|_{\mathcal{H}_1}, \ldots, \|h_p\|_{\mathcal{H}_p})\big)$ is a norm on $\mathcal{B}$ whose dual norm is $\Lambda^* : g \mapsto \Omega^*\big((\|g_1\|_{\mathcal{H}_1}, \ldots, \|g_p\|_{\mathcal{H}_p})\big)$. Moreover the subgradient of Λ is $\partial\Lambda(h) = A(h)$ with $A(h) := \{(u_1 s_1, \ldots, u_p s_p) \mid \boldsymbol{u} \in B(h),\ s_i \in \partial\|\cdot\|_{\mathcal{H}_i}(h_i)\}$ with $B(h) := \partial\Omega\big((\|h_1\|_{\mathcal{H}_1}, \ldots, \|h_p\|_{\mathcal{H}_p})\big)$.*

Proof It is clear that Λ is symmetric, positive and homogenous. The triangle inequality results from the fact that Ω is *monotonic*. Indeed the latter property implies that $\Lambda(h+g) = \Omega\big((\|h_i + g_i\|_{\mathcal{H}_i})_{1\leq i\leq p}\big) \leq \Omega\big((\|h_i\|_{\mathcal{H}_i} + \|g_i\|_{\mathcal{H}_i})_{1\leq i\leq p}\big)$ and the result follows from applying the triangle inequality for Ω.

Moreover, we have the generalized Cauchy-Schwarz inequality:

$$\langle h, g\rangle_{\mathcal{B}} = \sum_i \langle h_i, g_i\rangle_{\mathcal{H}_i} \leq \sum_i \|h_i\|_{\mathcal{H}_i}\|g_i\|_{\mathcal{H}_i} \leq \Lambda(h)\,\Lambda^*(g),$$

and it is easy to check that equality is attained if and only if $g \in A(h)$. This simultaneously shows that $\Lambda(h) = \max_{g\in\mathcal{B}}\langle h, g\rangle_{\mathcal{B}} - \Lambda^*(g)$ and that $\partial\Lambda(h) = A(h)$. ■

We consider now a learning problem of the form:

$$\min_{h\in\mathcal{B}} \psi(h_1, \ldots, h_p) + \lambda\Lambda(h) \tag{3.8}$$

with $\psi(h) = \frac{1}{n}\sum_{i=1}^n \ell(h(\boldsymbol{x}_i), \boldsymbol{y}_i)$. The structured MKL case corresponds more specifically to the case where $\psi(h) = \frac{1}{n}\sum_{i=1}^n \ell(h_1(\boldsymbol{x}_i) + \ldots + h_p(\boldsymbol{x}_p), \boldsymbol{y}_i)$. Note the problem we consider here is regularized with Λ and not Λ^2 as opposed to the formulations (1.21) and (1.28) considered in section 1.5.

To apply the proximal methods introduced in this chapter in $\mathcal{B}$ using $\|\cdot\|_{\mathcal{B}}$ as the proximal term requires to be able to solve the proximal problem:

$$\min_{h\in\mathcal{B}} \frac{1}{2}\|h - g\|_{\mathcal{B}}^2 + \mu\,\Lambda(h). \tag{3.9}$$

The following lemma shows that if we know how to compute the proximal operator of Ω for an ℓ_2 proximity term in $\mathbb{R}^p$, we can readily compute the proximal operator of Λ for the proximity defined with the Hilbert norm on $\mathcal{B}$:

Lemma 3.3 *$Prox_{\mu\Lambda}(g) = (y_1 s_1, \ldots, y_p s_p)$ where $s_i = 0$ if $g_i = 0$,*

$$s_i = \frac{g_i}{\|g_i\|_{\mathcal{H}_i}} \text{ if } g_i \neq 0 \quad \text{and} \quad y = Prox_{\mu\Omega}\big((\|g_i\|_{\mathcal{H}_i})_{1\leq i\leq p}\big).$$

Proof To lighten the notations, we write $\|h_i\|$ for $\|h_i\|_{\mathcal{H}_i}$ if $h_i \in \mathcal{H}_i$. The optimality condition for problem (3.9) is $h - g \in -\mu\partial\Lambda$ so that we have $h_i = g_i - \mu\, s_i u_i$, with $\boldsymbol{u} \in B(h)$, $s_i \in \partial\|\cdot\|_{\mathcal{H}_i}(h_i)$. The last equation implies that h_i, g_i and s_i are colinear. If $g_i = 0$ then the fact that Ω is *monotonic* implies that $h_i = s_i = 0$. If on the other hand, $g_i \neq 0$ we have $h_i = g_i\,(1 - \frac{\mu\, u_i}{\|g_i\|})_+$ and thus $\|h_i\| = (\|g_i\| - \mu\, u_i)_+$ and $h_i = s_i\,\|h_i\|$, but by definition of y_i we have $y_i = (\|g_i\| - \mu\, u_i)_+$, which shows the result. ■

This results shows how to compute the proximal operator at an abstract level. For the algorithm to be practical, we need to show that the corresponding computation can be performed by manipulating a finite number of parameters.

Fortunately, we can appeal to a representer theorem to that end, which leads to the following lemma

Lemma 3.4 *Assume that for all i, $g_i = \sum_{j=1}^n \boldsymbol{\alpha}_{ij} K_i(x_j, \cdot)$. Then the solution of problem (3.9) is of the form $h_i = \sum_{j=1}^n \boldsymbol{\beta}_{ij} K_i(x_j, \cdot)$. Let $\boldsymbol{y} = Prox_{\mu\Omega}\big((\sqrt{\boldsymbol{\alpha}_k^\top \boldsymbol{K}_k \boldsymbol{\alpha}_k})_{1\leq k\leq p}\big)$. Then if $\boldsymbol{\alpha}_i \neq 0$, $\boldsymbol{\beta}_i = \frac{\boldsymbol{y}_i}{\sqrt{\boldsymbol{\alpha}_i^\top \boldsymbol{K}_i \boldsymbol{\alpha}_i}} \boldsymbol{\alpha}_i$ and otherwise $\boldsymbol{\beta}_i = 0$.*

Proof We first show that a representer theorem holds. For each i let $h_i^{/\!/}$ be the component of h_i in the span of $(\boldsymbol{K}_i)(\boldsymbol{x}_j)_{1\leq j\leq n}$ and $h_i^\perp = h_i - h_i^{/\!/}$. We can rewrite the objective of problem (3.9) as[5]

$$\frac{1}{2}\sum_{i=1}^p \big[\|h_i^{/\!/}\|^2 + \|h_i^\perp\|^2 - 2\langle h_i^{/\!/}, g_i\rangle_{\mathcal{H}_i} + \|g_i\|^2\big] + \mu\Omega\big((\|h_i^{/\!/}\|^2 + \|h_i^\perp\|^2)_{1\leq i\leq p}\big)$$

from which, given that Ω is assumed *monotonic*, it is clear that setting $h_i^\perp = 0$ for all i can only decrease the objective. To conclude, the form of the solution in $\boldsymbol{\beta}$ results from the fact that $\|g_i\|_{\mathcal{H}_i}^2 = \sum_{1\leq j,j'\leq n} \boldsymbol{\alpha}_{ij}\boldsymbol{\alpha}_{ij'} \langle K_i(\boldsymbol{x}_j, \cdot), K_i(\boldsymbol{x}_{j'}, \cdot)\rangle_{\mathcal{H}_i}$ and $\langle K_i(\boldsymbol{x}_j, \cdot), K_i(\boldsymbol{x}_{j'}, \cdot)\rangle_{\mathcal{H}_i} = K_i(\boldsymbol{x}_j, \boldsymbol{x}_{j'})$ by the reproducing property, and by identification (Note that if the kernel matrix $\boldsymbol{K}_i$ is not invertible the solution might not be unique in $\boldsymbol{\beta}_i$). ■

Finally, in the last lemma we assumed that the function g_i in the proximal problem could be represented as a linear combination of the $\boldsymbol{K}_i(\boldsymbol{x}_j, \cdot)$. Since g_i is typically of the form $h_i^t - \frac{1}{L}\frac{\partial}{\partial h_i}\psi(h_1^t, \ldots, h_p^t)$, then the result follow by linearity from the fact that the gradient is in the span of the $\boldsymbol{K}_i(\boldsymbol{x}_j, \cdot)$. But we have the following lemma.

Lemma 3.5 *For $\psi(h) = \frac{1}{n}\sum_{j=1}^n \ell(h_1(\boldsymbol{x}_j), \ldots, h_p(\boldsymbol{x}_j), y_j)$ then*

$$\frac{\partial}{\partial h_i}\psi(h) = \sum_{j=1}^n \boldsymbol{\alpha}_i K_i(\boldsymbol{x}_j, \cdot) \quad \textit{for} \quad \boldsymbol{\alpha}_{ij} = \frac{1}{n}\partial_i \ell(h_1(\boldsymbol{x}_j), \ldots, h_p(\boldsymbol{x}_j), y_j),$$

where $\partial_i \ell$ denote the partial derivative of ℓ w.r.t. to its ith scalar component.

Proof This result follows from the rules of composition of differentiation applied to the functions

$$(h_1, \ldots, h_p) \mapsto \ell\big(\langle h_1, K_1(x_j, \cdot)\rangle_{\mathcal{H}_1}, \ldots, \langle h_p, K_p(x_j, \cdot)\rangle_{\mathcal{H}_p}, y_j\big)$$

and the fact that, since $h_i \mapsto \langle h_i, K_i(\boldsymbol{x}_j, \cdot)\rangle_{\mathcal{H}_i}$ is linear, its gradient in the RKHS $\mathcal{H}_i$ is just $K_i(\boldsymbol{x}_j, \cdot)$.

■

[5]We denote again $\|h_i\|$ for $\|h_i\|_{\mathcal{H}_i}$, when the RHKS norm used is implied by the argument.

Chapter 4

(Block) Coordinate Descent Algorithms

Coordinate descent algorithms solving ℓ_1-regularized learning problem go back to [47]. They optimize (exactly or approximately) the objective with respect to one variable at a time while all others are kept fixed. Note that, in general, coordinate descent algorithm are not necessarily convergent for non-smooth optimization; they are however applicable in this setting because of a *separability* property of the nonsmooth regularizer we consider (see end of Section 4.1).

4.1 Coordinate descent for ℓ_1-regularization

We consider first the following special case of one-dimensional ℓ_1-regularized problem:

$$\min_{w\in\mathbb{R}} \frac{1}{2}(w - w_0)^2 + \lambda|w|. \tag{4.1}$$

As shown in (1.5), $w^\star$ can be obtained by *soft-thresholding*:

$$w^\star = \mathrm{Prox}_{\lambda|\cdot|}(w_0) := \Big(1 - \frac{\lambda}{|w_0|}\Big)_+ w_0. \tag{4.2}$$

Lasso case. In the case of the least-square loss, the minimization with respect to a single coordinate can be written as

$$\min_{\boldsymbol{w}_j\in\mathbb{R}} \nabla_j f(\boldsymbol{w}^t)\,(\boldsymbol{w}_j - \boldsymbol{w}_j^t) + \frac{1}{2}\nabla^2_{jj} f(\boldsymbol{w}^t)(\boldsymbol{w}_j - \boldsymbol{w}_j^t)^2 + \lambda|\boldsymbol{w}_j|,$$

with $\nabla_j f(\boldsymbol{w}) = \boldsymbol{X}_j^T(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})$ and $\nabla^2_{jj} f(\boldsymbol{w}) = \boldsymbol{X}_j^T\boldsymbol{X}_j$ independent of $\boldsymbol{w}$. Since the above equation is of the form (4.1), it is solved in closed form:

$$\boldsymbol{w}_j^\star = \mathrm{Prox}_{\lambda|\cdot|}\big(\,\boldsymbol{w}_j^t - \nabla_j f(\boldsymbol{w}_j^t)/\nabla^2_{jj} f\,\big). \tag{4.3}$$

In other words, $\boldsymbol{w}_j^\star$ is obtained by solving the unregularized problem with respect to coordinate j and soft-thresholding the solution.

This is the update proposed in the shooting algorithm of Fu [47], which cycles through all variables in a fixed order.[1] Other cycling schemes are possible (see, e.g., [90]).

An efficient implementation is obtained if the quantity $\boldsymbol{X}\boldsymbol{w}^t - \boldsymbol{y}$ or even better $\nabla f(\boldsymbol{w}^t) = \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}^t - \boldsymbol{X}^T\boldsymbol{y}$ is kept updated.[2]

Smooth loss. For more general smooth losses, like the logistic loss, the optimization with respect to a single variable cannot be solved in closed form. It is possible to solve it numerically using a sequence of modified Newton steps as proposed by [110]. We present here a fast algorithm of Tseng and Yun [119] based on solving just a quadratic approximation of f with an inexact line search at each iteration.

Let $L^t > 0$ be a parameter and let $\boldsymbol{w}_j^\star$ be the solution of

$$\min_{\boldsymbol{w}_j \in \mathbb{R}} \nabla_j f(\boldsymbol{w}^t)\,(\boldsymbol{w}_j - \boldsymbol{w}_j^t) + \frac{1}{2}L^t\,(\boldsymbol{w}_j - \boldsymbol{w}_j^t)^2 + \lambda|\boldsymbol{w}_j|,$$

Given $d = \boldsymbol{w}_j^\star - \boldsymbol{w}_j^t$ where $\boldsymbol{w}_j^\star$ is the solution of (4.3), the algorithm of Tseng and Yun performs a line search to choose the largest step of the form αd with $\alpha = \alpha_0\beta^k$ and $\alpha_0 > 0, \beta \in (0,1)$, $k \in \mathbb{N}$, such that the following modified Armijo condition is satisfied:

$$F(\boldsymbol{w}^t + \alpha d\boldsymbol{e}_j) - F(\boldsymbol{w}^t) \leq \sigma\alpha\big(\nabla_j f(\boldsymbol{w})d + \gamma L^t d^2 + |\boldsymbol{w}_j^t + d| - |\boldsymbol{w}_j^t|\big),$$

where $F(\boldsymbol{w}) := f(\boldsymbol{w}) + \lambda\,\Omega(\boldsymbol{w})$, and $0 \leq \gamma \leq 1$ and $\sigma < 1$ are parameters of the algorithm.

Tseng and Yun [119] show that under mild conditions on f the algorithm is convergent and, under further assumptions, asymptotically linear. In particular, if f is of the form $\frac{1}{n}\sum_{i=1}^n \ell(y^{(i)}\boldsymbol{w}^\top\boldsymbol{x}^{(i)})$ with $\ell(y^i, \cdot)$ a twice continuously differentiable convex function with strictly positive curvature, the algorithm is asymptotically linear for $L^t = \nabla^2_{jj} f(\boldsymbol{w}_j^t)$. We refer the reader to section 4.2 and to [119, 126] for results under much milder conditions. It should be noted that the algorithm generalizes to other separable regularizations than the ℓ_1-norm.

Variants of coordinate descent algorithms have also been considered by [49], by [66], by [128]. Generalizations based on the Gauss-Southwell rule have been considered by [119].

Convergence of coordinate descent algorithms. In general, coordinate descent algorithms are not convergent for non-smooth objectives. Therefore, using such schemes always

[1] Coordinate descent with a cyclic order is sometimes called Gauss-Seidel procedure.

[2] In the former case, at each iteration, $\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}$ can be updated in $\Theta(n)$ operations if $\boldsymbol{w}_j$ changes and $\nabla_{j^{t+1}} f(\boldsymbol{w})$ can always be updated in $\Theta(n)$ operations. The complexity of one cycle is therefore $O(pn)$. However a better complexity is obtained in the latter case, provided the matrix $\boldsymbol{X}^T\boldsymbol{X}$ is precomputed (with complexity $O(p^2n)$). Indeed $\nabla f(\boldsymbol{w}^t)$ is updated in $\Theta(p)$ iterations only if w_j does not stay at 0. Otherwise, if w_j stays at 0 the step costs $O(1)$; the complexity of one cycle is therefore $\Theta(ps)$ where s is the number of non-zero variables at the end of the cycle.

require a convergence analysis. In the context of the ℓ_1-norm regularized smooth objective, the non-differentiability is *separable* (i.e., is a sum of non-differentiable terms that depend on single variables), and this is sufficient for convergence [18, 119]. In terms of convergence rates, coordinate descent behaves in a similar way than first-order methods such as proximal methods, i.e., if the objective function is strongly convex [90, 119], then the convergence is linear, while it is slower if the problem is not strongly convex, i.e., in the learning context, if there are strong correlations between input variables [107].

4.2 Block-coordinate descent for ℓ_1/ℓ_q regularization

When $\Omega(\boldsymbol{w})$ is the ℓ_1/ℓ_q-norm with groups $g \in \mathcal{G}$ forming a partition of $\{1, \dots, p\}$, the previous methods are generalized by block-coordinate descent (BCD) algorithms, that have been the focus of recent work by Tseng and Yun [119] and Wright [126]. These algorithms do not attempt to solve exactly a reduced problem on a block of coordinates but rather optimize a surrogate of F in which the function f is substituted a quadratic approximation.

Specifically, the BCD scheme of [119] solves at each iteration a problem of the form:

$$\min_{\boldsymbol{w}_g \in \mathbb{R}^{|g|}} \nabla_g f(\boldsymbol{w}^t)^T (\boldsymbol{w}_g - \boldsymbol{w}_g^t) + \frac{1}{2} (\boldsymbol{w}_g - \boldsymbol{w}_g^t)^T \boldsymbol{H}^t (\boldsymbol{w}_g - \boldsymbol{w}_g^t) + \lambda \|\boldsymbol{w}_g\|_q, \tag{4.4}$$

where the positive semi-definite matrix $\boldsymbol{H}^t \in \mathbb{R}^{|g| \times |g|}$ is a parameter. The above problem is solved in closed form if $\boldsymbol{H}^t = L^t \boldsymbol{I}_{|g|}$ for some scalar L^t and $q \in \{2, \infty\}$[3]. In particular for $q = 2$, the solution $\boldsymbol{w}_g^\star$ is obtained by *group-soft-thresholding*:

$$\boldsymbol{w}_g^\star = \mathrm{Prox}_{\frac{\lambda}{L^t} \|\cdot\|_2} \big(\boldsymbol{w}_g^t - \frac{1}{L^t} \nabla_g f(\boldsymbol{w}_g^t) \big),$$

with

$$\mathrm{Prox}_{\mu \|\cdot\|_2}(\boldsymbol{w}) = \Big(1 - \frac{\mu}{\|\boldsymbol{w}\|_2} \Big)_+ \boldsymbol{w}.$$

In the case of general smooth losses, the descent direction is given by $\boldsymbol{d} = \boldsymbol{w}_g^\star - \boldsymbol{w}_g^t$ with $\boldsymbol{w}_g^\star$ as above. The next point is of the form $\boldsymbol{w}^t + \alpha \boldsymbol{d}$, where α is a stepsize of the form $\alpha = \alpha_0 \, \beta^k$, with $\alpha_0 > 0$, $0 < \beta < 1$, $k \in \mathbb{N}$. k is chosen large enough to satisfy the following modified Armijo condition

$$F(\boldsymbol{w}^t + \alpha \boldsymbol{d}) - F(\boldsymbol{w}^t) \leq \sigma \alpha \big(\nabla_g f(\boldsymbol{w})^T \boldsymbol{d} + \gamma \, \boldsymbol{d}^\top \boldsymbol{H}^t X \boldsymbol{d} + \|\boldsymbol{w}_g^t + \boldsymbol{d}\|_q - \|\boldsymbol{w}_g^t\|_q \big),$$

for parameters $0 \leq \gamma \leq 1$ and $\sigma < 1$.

If f is convex continuously differentiable, lower bounded on $\mathbb{R}^p$ and F has a unique minimizer, provided that there exists $\tau, \bar{\tau}$ fixed constants such that for all t, $\tau \preceq H^t \preceq \bar{\tau}$ for all t, the results of Tseng and Yun show that the algorithm converges (see Theorem 4.1 in [119] for broader conditions). Wright [126] proposes a variant of the algorithm, in which the line-search on α is replaced by a line search on the parameter L^t, similar to the line-searches used in proximal methods.

[3] More generally for $q \geq 1$ and $\boldsymbol{H}^t = L^t \boldsymbol{I}_{|g|}$, it can be solved efficiently coordinate-wise using bisection algorithms.

4.3 Block-coordinate descent for MKL

Finally, block-coordinate descent algorithms are also applicable to classical multiple kernel learning (and also to all group Lasso formulations [131]). We consider the same setting and notations as in Section 3.4 and we consider specifically the optimization problem:

$$\min_{h \in \mathcal{B}} \psi(h_1, \dots, h_p) + \lambda \sum_{i=1}^{p} \|h_i\|_{\mathcal{H}_i}$$

A block-coordinate algorithm can be applied by considering each RKHS $\mathcal{H}_i$ as one "block"; this type of algorithm was considered by [99]. Applying the lemmas 3.4 and 3.5 of section 3.4, we know that h_i can be represented as $h_i = \sum_{j=1}^{n} \boldsymbol{\alpha}_{ij} K_i(\boldsymbol{x}_j, \cdot)$.

The algorithm then consists in performing successively group soft-thresholding in each RKHS $\mathcal{H}_i$. This can be done by working directly with the dual parameters $\boldsymbol{\alpha}_i$, with a corresponding proximal operator in the dual simply formulated as:

$$\mathrm{Prox}_{\mu \|\cdot\|_{\boldsymbol{K}_i}}(\boldsymbol{\alpha}_i) = \Big(1 - \frac{\mu}{\|\boldsymbol{\alpha}_i\|_{\boldsymbol{K}_i}}\Big)_+ \boldsymbol{\alpha}_i.$$

with $\|\boldsymbol{\alpha}\|_{\boldsymbol{K}}^2 = \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{\alpha}$.

Chapter 5

Reweighted-ℓ_2 Algorithms

Approximating a nonsmooth or constrained optimization problem by a series of smooth unconstrained problems is common in optimization (see, e.g., [25, 88, 91]). In the context of objective functions regularized by sparsity-inducing norms, it is natural to consider variational formulations of these norms in terms of squared ℓ_2-norms, since many efficient methods are available to solve ℓ_2-regularized problems (e.g., linear system solvers for least-squares regression).

In this section, we show on our motivating example of sums of ℓ_2-norms of subsets how such formulations arise (see, e.g., [37, 60, 95, 96]).

The variational formulation we have presented in the previous proposition allows to consider the following function $H(\boldsymbol{w}, \boldsymbol{\eta})$ defined as

$$H(\boldsymbol{w}, \boldsymbol{\eta}) = f(\boldsymbol{w}) + \frac{\lambda}{2} \sum_{j=1}^{p} \Big\{ \sum_{g \in \mathcal{G}, j \in g} \boldsymbol{\eta}_g^{-1} \Big\} \boldsymbol{w}_j^2 + \frac{\lambda}{2} \sum_{g \in \mathcal{G}} \boldsymbol{\eta}_g.$$

It is jointly convex in $(\boldsymbol{w}, \boldsymbol{\eta})$; the minimization with respect to $\boldsymbol{\eta}$ can be done in closed form, and the optimum is equal to $F(\boldsymbol{w}) = f(\boldsymbol{w}) + \lambda \Omega(\boldsymbol{w})$; as for the minimization with respect to $\boldsymbol{w}$, it is a ℓ_2-regularized problem.

Unfortunately, the alternating minimization algorithm that is immediately suggested is not convergent in general, because the function H is not continuous (in particular around $\boldsymbol{\eta}$ which has zero coordinates). In order to make the algorithm convergent, two strategies are commonly used:

- **Smoothing**: we can add a term of the form $\frac{\varepsilon}{2} \sum_{g \in \mathcal{G}} \boldsymbol{\eta}_g^{-1}$, which yields a joint cost function with compact level sets on the set of positive numbers. Alternating minimization algorithms are then convergent (as a consequence of general results on block coordinate descent), and have two different iterations: (1) minimization with respect to $\boldsymbol{\eta}$ in closed form, through $\boldsymbol{\eta}_g = \sqrt{\|\boldsymbol{w}_g\|_2^2 + \varepsilon}$, and (2) minimization with respect to $\boldsymbol{w}$, which is an ℓ_2-regularized problem, which can be for example solved in closed form for the square loss. Note however, that the second problem need not be optimized exactly at all iterations.

- **First order method in $\boldsymbol{\eta}$**: While the joint cost function $H(\boldsymbol{\eta}, \boldsymbol{w})$ is not continuous, the function $I(\boldsymbol{\eta}) = \min_{\boldsymbol{w} \in \mathbb{R}^p} H(\boldsymbol{w}, \boldsymbol{\eta})$ is continuous, and under general assumptions, continuously differentiable, and is thus amenable to first-order methods (e.g., proximal methods, gradient descent). When the groups in $\mathcal{G}$ do not overlap, one sufficient condition is that the function $f(\boldsymbol{w})$ is of the form $f(\boldsymbol{w}) = \psi(\boldsymbol{X}\boldsymbol{w})$ for $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ any matrix (typically the design matrix) and ψ a strongly convex function on $\mathbb{R}^n$. This strategy is particularly interesting when evaluating $I(\boldsymbol{\eta})$ is computationally cheap.

In theory, the alternating scheme consisting in optimizing alternatively over $\boldsymbol{\eta}$ and $\boldsymbol{w}$ can be used to solve learning problems regularized with *any* norms: we indeed show in the next section that any norm admits a quadratic variational formulation. To illustrate the principle of ℓ_2-reweighted algorithms, we first consider the special case of multiple kernel learning; in Section 5.1, we consider the case of the trace norm.

Structured MKL. Reweighted-ℓ_2 algorithms are fairly natural for norms which admit a diagonal variational formulation (see Lemma 1.2 and [81]) and for the corresponding multiple kernel learning problem. We consider the structured multiple learning problem presented in Section 1.5.4.

The alternating scheme applied to equation (1.27) then takes the following form: for $\boldsymbol{\eta}$ fixed, one has to solve a single kernel learning problem with the kernel $K = \sum_i \boldsymbol{\eta}_i K_i$; the corresponding solution in the product of RKHSes $\mathcal{H}_1 \times \ldots \times \mathcal{H}_p$ (see section 3.4) is of the form $h(\boldsymbol{x}) = h_1(\boldsymbol{x}) + \ldots + h_p(\boldsymbol{x})$ with $h_i(\boldsymbol{x}) = \boldsymbol{\eta}_i \sum_{j=1}^n \boldsymbol{\alpha}_j K_i(\boldsymbol{x}_j, \cdot)$. Since $\|h_i\|_{\mathcal{H}_i}^2 = \boldsymbol{\eta}_i^2 \boldsymbol{\alpha}^\top \boldsymbol{K}_i \boldsymbol{\alpha}$, for fixed $\boldsymbol{\alpha}$, the update in $\boldsymbol{\eta}$ then takes the form:

$$\boldsymbol{\eta}^{t+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{\eta} \in H} \sum_{i=1}^p \frac{(\boldsymbol{\eta}_i^t)^2 {\boldsymbol{\alpha}^t}^\top \boldsymbol{K}_i \, \boldsymbol{\alpha}^t + \varepsilon}{\boldsymbol{\eta}_i}.$$

Note that these updates produce a non-increasing sequence of values of the primal objective. Moreover, this MKL optimization scheme uses a potentially much more compact parameterization than proximal methods since in addition to the variational parameter $\boldsymbol{\eta} \in \mathbb{R}^p$ a single vector of parameter $\boldsymbol{\alpha} \in \mathbb{R}^n$ is needed as opposed to up to one such vector for each kernel in the case of proximal methods. MKL problems can also be tackled using first order methods in $\boldsymbol{\eta}$ described above: we refer the reader to [96] for an example in the case of classical MKL.

5.1 Quadratic variational formulation for general norms

We now investigate a general variational formulation of norms that naturally leads to a sequence of reweighted ℓ_2-regularized problems. The formulation is based on approximating the unit ball of a norm Ω with enclosing ellipsoids. See Figure 5.1. The following proposition shows that all norms may be expressed as a minimum of Euclidean norms:

Proposition 5.1 *Let $\Omega : \mathbb{R}^p \to \mathbb{R}$ a norm on $\mathbb{R}^p$, then there exists a function g defined on the cone of positive semi-definite matrices $\mathbf{S}_p^+$, such that g is convex, strictly positive except*

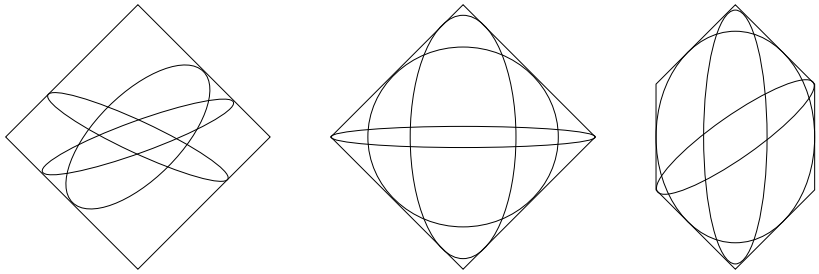


Figure 5.1: Example of a sparsity-inducing ball in two dimensions, with enclosing ellipsoids. Left: ellipsoids with general axis for the ℓ_1-norm, middle: ellipsoids with horizontal and vertical axis for the ℓ_1-norm, right: ellipsoids for another polyhedral norm.

at zero, positively homogeneous and such that

$$\forall \boldsymbol{w} \in \mathbb{R}^p,\ \Omega(\boldsymbol{w}) = \min_{\boldsymbol{\Lambda} \in \mathbf{S}_p^+,\ g(\boldsymbol{\Lambda}) \leqslant 1} \sqrt{\boldsymbol{w}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{w}} = \frac{1}{2} \min_{\boldsymbol{\Lambda} \in \mathbf{S}_p^+} \big\{ \boldsymbol{w}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{w} + g(\boldsymbol{\Lambda}) \big\}. \tag{5.1}$$

Proof Let Ω^* be the dual norm of Ω, defined as $\Omega^*(\boldsymbol{s}) = \max_{\Omega(\boldsymbol{w}) \leqslant 1} \boldsymbol{w}^T \boldsymbol{s}$ [25]. Let g be the function defined through $g(\boldsymbol{\Lambda}) = \max_{\Omega^*(\boldsymbol{s}) \leqslant 1} \boldsymbol{s}^T \boldsymbol{\Lambda} \boldsymbol{s}$. This function is well-defined as the maximum of a continuous function over a compact set; moreover, as a maximum of linear functions, it is convex and positive homogeneous. Also, for nonzero $\boldsymbol{\Lambda}$, the quadratic form $\boldsymbol{s} \mapsto \boldsymbol{s}^T \boldsymbol{\Lambda} \boldsymbol{s}$ is not identically zero around $\boldsymbol{s} = 0$, hence the strict positivity of g.

Let $\boldsymbol{w} \in \mathbb{R}^p$ and $\boldsymbol{\Lambda} \in \mathbf{S}_p^+$; there exists $\boldsymbol{s}$ such that $\Omega^*(\boldsymbol{s}) = 1$ and $\boldsymbol{w}^T \boldsymbol{s} = \Omega(\boldsymbol{w})$. We then have

$$\Omega(\boldsymbol{w})^2 = (\boldsymbol{w}^T \boldsymbol{s})^2 \leqslant (\boldsymbol{w}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{w})(\boldsymbol{s}^T \boldsymbol{\Lambda} \boldsymbol{s}) \leqslant g(\boldsymbol{\Lambda})(\boldsymbol{w}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{w}).$$

This shows that $\Omega(\boldsymbol{w}) \leqslant \min_{\boldsymbol{\Lambda} \in \mathbf{S}_p^+,\ g(\boldsymbol{\Lambda}) \leqslant 1} \sqrt{\boldsymbol{w}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{w}}$. Proving the other direction can be done using the following limiting argument. Given $\boldsymbol{w}_0 \in \mathbb{R}^p$, consider $\boldsymbol{\Lambda}(\varepsilon) = (1 - \varepsilon) \boldsymbol{w}_0 \boldsymbol{w}_0^T + \varepsilon (\boldsymbol{w}_0^T \boldsymbol{w}_0) I$. We have $\boldsymbol{w}_0^T \boldsymbol{\Lambda}(\varepsilon)^{-1} \boldsymbol{w}_0 = 1$ and $g(\boldsymbol{\Lambda}(\varepsilon)) \to g(\boldsymbol{w}_0 \boldsymbol{w}_0^T) = \Omega(\boldsymbol{w}_0)^2$. Thus, for $\tilde{\boldsymbol{\Lambda}}(\varepsilon) = \boldsymbol{\Lambda}(\varepsilon) / g(\boldsymbol{\Lambda}(\varepsilon))$, we have that $\sqrt{\boldsymbol{w}_0^T \tilde{\boldsymbol{\Lambda}}(\varepsilon)^{-1} \boldsymbol{w}_0}$ tends to $\Omega(\boldsymbol{w}_0)$, thus $\Omega(\boldsymbol{w}_0)$ must be no smaller than the minimum over all $\boldsymbol{\Lambda}$. The right-hand side of Eq. (5.1) can be obtained by optimizing over the scale of $\boldsymbol{\Lambda}$. ■

Note that while the proof provides a closed-form expression for a candidate function g, it is not unique, as can be seen in the following examples, the domain of g (matrices so that g is finite) may be reduced (in particular to diagonal matrices for the ℓ_1-norm and more generally the sub-quadratic norms defined in Section 1.5.3):

- For the ℓ_1-norm: $g(\boldsymbol{\Lambda}) = \|\boldsymbol{\Lambda}^{1/2}\|_{1/2}^2$, where $\boldsymbol{\Lambda}^{1/2}$ is the positive square root of $\boldsymbol{\Lambda}$, but we could use $g(\boldsymbol{\Lambda}) = \mathrm{tr} \boldsymbol{\Lambda}$ if $\boldsymbol{\Lambda}$ is diagonal and $+\infty$ otherwise.
- For subquadratic norms (Section 1.5.3), we can take $g(\boldsymbol{\Lambda})$ to be zero for non-diagonal $\boldsymbol{\Lambda}$, and equal to the gauge function of the set H defined in Section 1.5.3, which

is $\bar{\Omega}$, applied to the diagonal of $\boldsymbol{\Lambda}$. For all of these norms, it is straightforward to apply to structured multiple kernel learning, i.e., following Section 1.5, replacing single variables by kernel matrices. In this situation, the ℓ_2-regularized problems may be solved using the kernel trick.

- For the ℓ_2-norm: $g(\boldsymbol{\Lambda}) = \lambda_{\max}(\boldsymbol{\Lambda})$ but we could of course use $g(\boldsymbol{\Lambda}) = 1$ if $\boldsymbol{\Lambda} = I$ and $+\infty$ otherwise.

- For the trace norm: $\boldsymbol{w}$ is assumed to be of the form $\boldsymbol{w} = \text{vect}(\boldsymbol{W})$ and the trace norm of $\boldsymbol{W}$ is regularized. We consider this case in more details.

  The trace norm admits the variational form (see [6]) :

$$\|\boldsymbol{W}\|_* = \frac{1}{2} \min_{\boldsymbol{D} \succeq 0} \text{tr}(\boldsymbol{W}^\top \boldsymbol{D}^{-1} \boldsymbol{W} + \boldsymbol{D}) \quad \text{s.t.} \quad \boldsymbol{D} \succ 0. \tag{5.2}$$

  But $\text{tr}(\boldsymbol{W}^\top \boldsymbol{D}^{-1} \boldsymbol{W}) = \boldsymbol{w}^\top (\boldsymbol{I} \otimes \boldsymbol{D})^{-1} \boldsymbol{w}$, which shows that the regularization by the trace norm takes the form of equation (5.1) in which we can choose $g(\boldsymbol{\Lambda})$ equal to $\text{tr}(\boldsymbol{D})$ if $\boldsymbol{\Lambda} = \boldsymbol{I} \otimes \boldsymbol{D}$ for some $\boldsymbol{D} \succ 0$ and $+\infty$ otherwise.

  The solution of the above optimization problem is given by $\boldsymbol{D}^* = (\boldsymbol{W}\boldsymbol{W}^\top)^{1/2}$ which can be computed via a singular value decomposition of $\boldsymbol{W}$.

  The reweighted-ℓ_2 algorithm to solve

$$\min_{\boldsymbol{W} \in \mathbb{R}^{p \times k}} f(\boldsymbol{W}) + \lambda \|\boldsymbol{W}\|_*$$

  therefore consists in iterating between the two updates (see, e.g., [6] for more details):

$$\boldsymbol{W} \leftarrow \operatorname*{argmin}_{\boldsymbol{W}} f(\boldsymbol{W}) + \frac{\lambda}{2} \text{tr}(\boldsymbol{W}^\top \boldsymbol{D}^{-1} \boldsymbol{W}) \quad \text{and}$$

$$\boldsymbol{D} \leftarrow (\boldsymbol{W}\boldsymbol{W}^\top + \varepsilon \boldsymbol{I}_k)^{1/2}$$

  where ε is a smoothing parameter that arises from adding a term $\frac{\varepsilon\lambda}{2} \text{tr}(\boldsymbol{D}^{-1})$ to Eq. (5.2) and prevents the matrix from becoming singular.

Chapter 6

Working-Set and Homotopy Methods

In this section, we consider methods that explicitly take into account the fact that the solutions are sparse, namely working set methods and homotopy methods.

6.1 Working-Set Techniques

Working-set algorithms address optimization problems by solving an increasing sequence of small subproblems of (1.1). The working set, that we will denote J, refers to the subset of variables involved in the optimization of these subproblems.

Working-set algorithms proceed as follows: after computing a solution to the problem restricted to the variables in J, global optimality is checked to determine whether the algorithm has to continue. If this is the case, new variables enter the working set J according to a strategy that has to be defined. Note that we only consider *forward* algorithms, i.e., where the working set grows monotonically. In other words, there are no *backward* steps where variables would be allowed to leave the set J. Provided this assumption is met, it is easy to see that these procedures stop in a finite number of iterations.

This class of algorithms is typically applied to linear programming and quadratic programming problems (see, e.g., [91]), and here takes specific advantage of sparsity from a computational point of view [9, 56, 69, 92, 102, 104, 113], since the subproblems that need to be solved are typically much smaller than the original one.

Working-set algorithms require three ingredients:

- **Inner-loop solver**: At each iteration of the working-set algorithm, problem (1.1) has to be solved on J, i.e., subject to the additional equality constraint that $\boldsymbol{w}_j = 0$ for all j in J^c:

$$\min_{\boldsymbol{w} \in \mathbb{R}^p} f(\boldsymbol{w}) + \lambda \Omega(\boldsymbol{w}), \text{ such that } \boldsymbol{w}_{J^c} = 0. \tag{6.1}$$

The computation can be performed by any of the methods presented in this chapter. Working-set algorithms should therefore be viewed as "meta-algorithms". Since solutions for successive working sets are typically close to each other the approach is efficient if the method chosen can use *warm-restarts*.

- **Computing the optimality conditions**: Given a solution $\boldsymbol{w}^\star$ of problem (6.1), it is then necessary to check whether $\boldsymbol{w}^\star$ is also a solution for the original problem (1.1). This test relies on the duality gaps of problems (6.1) and (1.1). In particular, if $\boldsymbol{w}^\star$ is a solution of problem (6.1), it follows from Proposition 1.3 in Section 1.4 that

$$f(\boldsymbol{w}^\star) + \lambda\Omega(\boldsymbol{w}^\star) + f^*(\nabla f(\boldsymbol{w}^\star)) = 0.$$

  In fact, the Lagrangian parameter associated with the equality constraint ensures the feasibility of the dual variable formed from the gradient of f at $\boldsymbol{w}^\star$. In turn, this guarantees that the duality gap of problem (6.1) vanishes. The candidate $\boldsymbol{w}^\star$ is now a solution of the full problem (1.1), i.e., without the equality constraint $\boldsymbol{w}_{J^c} = 0$, if and only if

$$\Omega^*(\nabla f(\boldsymbol{w}^\star)) \leq \lambda. \tag{6.2}$$

  Condition (6.2) points out that the dual norm Ω^* is a key quantity to monitor the progress of the working-set algorithm [56]. In simple settings, for instance when Ω is the ℓ_1-norm, checking condition (6.2) can be easily computed since Ω^* is just the ℓ_∞-norm. In this case, condition (6.2) becomes

$$|[\nabla f(\boldsymbol{w}^\star)]_j| \leq \lambda, \text{ for all } j \text{ in } \{1, \ldots, p\}.$$

  Note that by using the optimality of problem (6.1), the components of the gradient of f indexed by J are already guaranteed to be no greater than λ.

  For more general sparsity-inducing norms with overlapping groups of variables (see Section 1.3), the dual norm Ω^* cannot be computed easily anymore, prompting the need for approximations and upper-bounds of Ω^* [9, 56, 104].

- **Strategy for the growth of the working set**: If condition (6.2) is not satisfied for the current working set J, some inactive variables in J^c have to become active. This point raises the questions of *how many* and *how* these variables should be chosen.

  First, depending on the structure of Ω, a *single* or a *group* of inactive variables have to be considered to enter the working set. Furthermore, one natural way to proceed is to look at the variables that violate condition (6.2) most. In the example of ℓ_1-regularized least squares regression with normalized predictors, this strategy amounts to selecting the inactive variable that has the highest correlation with the current residual.

The working-set algorithms we have described so far aim at solving problem (1.1) for a fixed value of the regularization parameter λ. However, for specific types of loss and regularization functions, the set of solutions of problem (1.1) can be obtained efficiently for all possible values of λ, which is the topic of the next section.

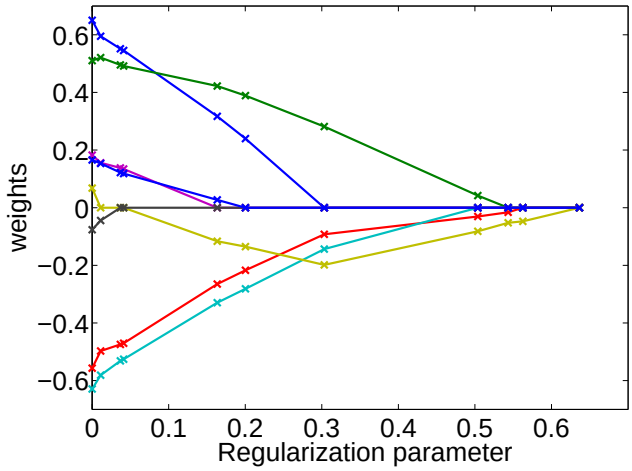


Figure 6.1: The weights $\boldsymbol{w}^\star(\lambda)$ are represented as functions of the regularization parameter λ. When λ increases, more and more coefficients are set to zero. These functions are all piecewise affine. Note that some variables (here one) may enter and leave the regularization path.

6.2 Homotopy methods

We present in this section an active-set[1] method for solving the Lasso problem [114] of Eq. (1.8). We recall the Lasso formulation:

$$\min_{\boldsymbol{w}\in\mathbb{R}^p} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda\|\boldsymbol{w}\|_1, \tag{6.3}$$

where $\boldsymbol{y}$ is in $\mathbb{R}^n$, and $\boldsymbol{X}$ is a design matrix in $\mathbb{R}^{n\times p}$. Even though generic working-set methods introduced above could be used to solve this formulation, a specific property of the ℓ_1-norm associated with a quadratic loss makes it possible to address it more efficiently.

Under mild assumptions (which we will detail later), the solution of Eq. (6.3) is unique, and we denote it by $\boldsymbol{w}^\star(\lambda)$. We call *regularization path* the function $\lambda \mapsto \boldsymbol{w}^\star(\lambda)$ that associates to a regularization parameter λ the corresponding solution. We will show that this function is piecewise linear, a behavior illustrated in Figure 6.1, where the entries of $\boldsymbol{w}^\star(\lambda)$ for a particular instance of the Lasso are represented as functions of λ.

An efficient algorithm can thus be constructed by chosing a particular value of λ, for which finding this solution is trivial, and by following the piecewise affine path, computing the directions of the current affine parts, and the points where the direction changes (also known as kinks). This piecewise linearity was first discovered and exploited by [79] in the context of portfolio selection, revisited by [94] describing an *homotopy* algorithm, and studied

[1] Active-set and working-set methods are very similar; their differ in that active-set methods allow (or sometimes require) variables returning to zero to exit the set.

by [40] with the LARS algorithm.[2] Similar ideas also appear early in the optimization literature: Finding the full regularization path of the Lasso is in fact a particular instance of a *parametric quadratic programming* problem, for which path following algorithms have been developed [100].

Let us show how to construct the path. From the optimality conditions we have presented in Eq. (1.9), denoting by $J := \{j; |\boldsymbol{X}_j^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}^\star)| = \lambda\}$ the set of active variables, and defining the vector $\boldsymbol{t}$ in $\{-1; 0; 1\}^p$ as $\boldsymbol{t} := \operatorname{sgn}\big(\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}^\star)\big)$, we have the following closed-form expression

$$\begin{cases} \boldsymbol{w}_J^\star(\lambda) & = (\boldsymbol{X}_J^T\boldsymbol{X}_J)^{-1}(\boldsymbol{X}_J^T\boldsymbol{y} - \lambda\boldsymbol{t}_J) \\ \boldsymbol{w}_{J^c}^\star(\lambda) & = 0, \end{cases}$$

where we have assumed the matrix $\boldsymbol{X}_J^T\boldsymbol{X}_J$ to be invertible (which is a sufficient condition to guarantee the uniqueness of $\boldsymbol{w}^\star$). This is an important point: if one knows in advance the set J and the signs $\boldsymbol{t}_J$, then $\boldsymbol{w}^\star(\lambda)$ admits a simple closed-form. Moreover, when J and $\boldsymbol{t}_J$ are fixed, the function $\lambda \mapsto (\boldsymbol{X}_J^T\boldsymbol{X}_J)^{-1}(\boldsymbol{X}_J^T\boldsymbol{y} - \lambda\boldsymbol{t}_J)$ is affine in λ. With this observation in hand, we can now present the main steps of the path-following algorithm. It basically starts from a trivial solution of the regularization path, follows the path by exploiting this formula, updating J and $\boldsymbol{t}_J$ whenever needed so that optimality conditions (1.9) remain satisfied. This procedure requires some assumptions—namely that (a) the matrix $\boldsymbol{X}_J^T\boldsymbol{X}_J$ is always invertible, and (b) that updating J along the path consists of adding or removing from this set a single variable at the same time. Concretely, we proceed as follows:

1. Set λ to $\|\boldsymbol{X}^T\boldsymbol{y}\|_\infty$ for which it is easy to show from Eq. (1.9) that $\boldsymbol{w}^\star(\lambda) = 0$ (trivial solution on the regularization path).
2. Set $J := \{j; |\boldsymbol{X}_j^T\boldsymbol{y}| = \lambda\}$.
3. Follow the regularization path by decreasing the value of λ, with the formula $\boldsymbol{w}_J^\star(\lambda) = (\boldsymbol{X}_J^T\boldsymbol{X}_J)^{-1}(\boldsymbol{X}_J^T\boldsymbol{y} - \lambda\boldsymbol{t}_J)$ keeping $\boldsymbol{w}_{J^c}^\star = 0$, until one of the following events occur
   - There exists j in J^c such that $|\boldsymbol{X}_j^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}^\star)| = \lambda$. Then, add j to the set J.
   - There exists j in J such that a non-zero coefficient $\boldsymbol{w}_j^\star$ hits zero. Then, remove j from J.

   We suppose that only one of such events can occur at the same time (b). It is also easy to show that the value of λ corresponding to the next event can be obtained in closed form.
4. Go back to 3.

Let us now briefly discuss assumptions (a) and (b). When the matrix $\boldsymbol{X}_J^T\boldsymbol{X}_J$ is not invertible, the regularization path is non-unique, and the algorithm fails. This can easily be fixed

[2] Even though the basic version of LARS is a bit different from the procedure we have just described, it is closely related, and indeed a simple modification makes it possible to obtain the full regularization path of Eq. (1.8).

by addressing instead a slightly modified formulation. It is possible to consider instead the elastic-net formulation of [134] that uses $\Omega(\boldsymbol{w}) = \lambda\|\boldsymbol{w}\|_1 + \frac{\gamma}{2}\|\boldsymbol{w}\|_2^2$. Indeed, it amounts to replacing the matrix $\boldsymbol{X}_J^T\boldsymbol{X}_J$ by $\boldsymbol{X}_J^T\boldsymbol{X}_J + \gamma\boldsymbol{I}$, which is positive definite and therefore always invertible, and to apply the same algorithm. The second assumption (b) can be unsatisfied in practice because of the machine precision. To the best of our knowledge, the algorithm will fail in such cases, but we consider this scenario unlikely with real data, though possible when the Lasso/basis pursuit is used multiple times such as in dictionary learning, presented in Section 7.3. In such situations, a proper use of optimality conditions can detect such problems and more stable algorithms such as proximal methods may then be used.

The complexity of the above procedure depends on the number of kinks of the regularization path (which is also the number of iterations of the algorithm). Even though it is possible to build examples where this number is large, we often observe in practice that the event where one variable gets out of the active set is rare. The complexity also depends on the implementation. By maintaining the computations of $\boldsymbol{X}_j^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}^\star)$ and a Cholesky decomposition of $(\boldsymbol{X}_J^T\boldsymbol{X}_J)^{-1}$, it is possible to obtain an implementation in $O(psn + ps^2 + s^3)$ operations, where s is the sparsity of the solution when the algorithm is stopped (which we approximately consider as equal to the number of iterations). The product psn corresponds to the computation of the matrices $\boldsymbol{X}_J^T\boldsymbol{X}_J$, ps^2 to the updates of the correlations $\boldsymbol{X}_j^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}^\star)$ along the path, and s^3 to the Cholesky decomposition.

Chapter 7

Sparsity and Nonconvex Optimization

In this section, we consider alternative approaches to sparse modelling, which are not based in convex optimization, but often use convex optimization problems in inner loops.

7.1 Greedy Algorithms

First, we consider the following ℓ_0-constrained signal decomposition problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^p} \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w}\|_2^2 \text{ s.t. } \|\boldsymbol{w}\|_0 \leq s, \tag{7.1}$$

where s is the desired sparsity of the solution, and we assume for simplicity that the columns of $\boldsymbol{X}$ have unit norm. Even though this problem can be shown to be NP-hard [84], greedy procedures can provide an approximate solution. Under some assumptions on the matrix $\boldsymbol{X}$, they can also be shown to have some optimality guarantees [116].

Several variants of these algorithms with different names have been developed both by the statistics and signal processing communities. In a nutshell, they are known as forward selection techniques in statistics (see [124]), and matching pursuit algorithms in signal processing [78]. All of these approaches start with a null vector $\boldsymbol{w}$, and iteratively increase the sparsity of $\boldsymbol{w}$ until it reaches the threshold s.

The algoritm dubbed *matching pursuit*, was introduced in the 90's in [78], and can be seen as a non-cyclic coordinate descent procedure for minimizing Eq. (7.1). It selects at each step a column $\boldsymbol{x}^{\hat{\imath}}$ that is the most correlated with the residual according to the formula

$$\hat{\imath} \leftarrow \operatorname*{argmin}_{i\in[1;p]} |\boldsymbol{r}^\top \boldsymbol{x}^i|,$$

where $\boldsymbol{r}$ denotes the residual $\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w}$. Then, the residual is projected on $\boldsymbol{x}^{\hat{\imath}}$ and the entry $\boldsymbol{w}_{\hat{\imath}}$ is updated according to

$$\begin{aligned}\boldsymbol{w}_{\hat{\imath}} &\leftarrow \boldsymbol{w}_{\hat{\imath}} + \boldsymbol{r}^\top \boldsymbol{x}^{\hat{\imath}} \\ \boldsymbol{r} &\leftarrow \boldsymbol{r} - (\boldsymbol{r}^\top \boldsymbol{x}^{\hat{\imath}})\boldsymbol{x}^{\hat{\imath}}.\end{aligned}$$

Such a simple procedure is guaranteed to decrease the objective function at each iteration, but is not to converge in a finite number of steps (the same variable can be selected several times during the process). Note that such a scheme also appears in statistics in boosting procedures [46].

Orthogonal matching pursuit was proposed as a major variant of matching pursuit that ensures the residual of the decomposition to be always *orthogonal to all previously selected columns of* $\boldsymbol{X}$. Such technique existed in fact in the statistics literature under the name *forward selection* [124], and a particular implementation exploiting a QR matrix factorization also appears early in [84]. More precisely, the algorithm is an active set procedure, which sequentially adds one variable at a time to the active set, which we denote by J. It provides an approximate solution of Eq. (7.1) for every sparsity value $s' \leq s$, and stops when the desired sparsity is reached. Thus, it builds a regularization path, and shares many similarities with the homotopy algorithm for solving the Lasso [40], even though the two algorithms address different optimization problems. These similarities are also very strong in terms of implementation: Identical tricks as those described in Section 6.2 for the homotopy algorithm can be used, and in fact both algorithms have roughly the same complexity (if most variables do not leave the path once they have entered), and have many steps in common. At each iteration, one has to choose which new predictor should enter the active set J. A possible choice is to look for the column of $\boldsymbol{X}$ most correlated with the residual as in the matching pursuit algorithm, but another criterion is to select the one that helps most reducing the objective function

$$\hat{\imath} \leftarrow \operatorname*{argmin}_{i \in J^c} \min_{\boldsymbol{w}' \in \mathbb{R}^{|J|+1}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}_{J\cup\{i\}}\boldsymbol{w}'\|_2^2.$$

Whereas this choice seem at first sight computationally expensive since it requires solving $|J^c|$ least-squares problems, the solution can in fact be obtained efficiently using a few tricks, based on Cholesky matrix decomposition and basic linear algebra, which we will not detail here for simplicity reasons (see [36] for more details).

After this step, the active set is updated $J \leftarrow J \cup \{\hat{\imath}\}$, and the corresponding residual $\boldsymbol{r}$ and coefficients $\boldsymbol{w}$ are

$$\begin{aligned}\boldsymbol{w} &\leftarrow (\boldsymbol{X}_J^\top \boldsymbol{X}_J)^{-1}\boldsymbol{X}_J^\top \boldsymbol{y}, \\ \boldsymbol{r} &\leftarrow (\boldsymbol{I}_p - \boldsymbol{X}_J(\boldsymbol{X}_J^\top \boldsymbol{X}_J)^{-1}\boldsymbol{X}_J^\top)\boldsymbol{y},\end{aligned}$$

where $\boldsymbol{r}$ is the residual of the orthogonal projection of $\boldsymbol{y}$ onto the linear subspace spanned by the columns of $\boldsymbol{X}_J$. It is worth noticing that one does not need to compute these two quantities in practice, but only updating the Cholesky decomposition of $(\boldsymbol{X}_J^\top \boldsymbol{X}_J)^{-1}$ and computing directly $\boldsymbol{X}^\top \boldsymbol{r}$, via simple linear algebra relations.

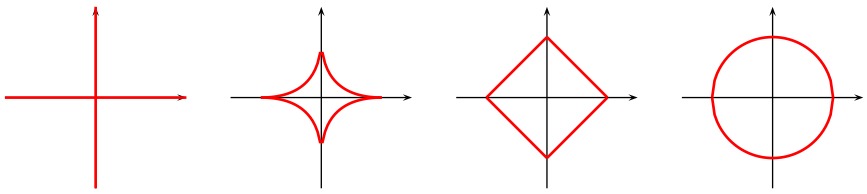
These greedy algorithms for solving the ℓ_0 sparse approximation problem admit several extensions when the regularization is more complex than the ℓ_0 pseudo-norm. For instance, they are used in the context of non-convex group-sparsity in [117], or with structured sparsity formulations [15, 53].

Other possibilities than greedy methods exists for optimizing Eq. (7.1). For instance, one can use the algorithm ISTA (i.e., the non-accelerated proximal method) presented in Section 3 when the function f is convex and its gradient Lipshitz continuous. Under this assumption, it is easy to see that ISTA can iteratively decrease the value of the nonconvex objective function. Such proximal gradient algorithms when Ω is the ℓ_0 pseudo-norm often appear under the name of iterative hard-thresholding methods [51].

7.2 DC-Programming, Reweighted-ℓ_1 Algorithms

We present in this section optimization schemes dedicated to particular non-convex regularization functions. More precisely, we address problem (7.1) but we now consider Ω which is a nonconvex penalty that is separable and takes the form $\Omega(\boldsymbol{w}) := \sum_{i=1}^{p} \psi(|\boldsymbol{w}_i|)$, where $\boldsymbol{w}$ is in $\mathbb{R}^p$, and $\psi : \mathbb{R} \to \mathbb{R}^+$ is a *concave* non-decreasing differentiable function. Examples of such penalties include variants of ℓ_q-penalties for $q < 1$ defined as $\psi : t \mapsto (|t| + \varepsilon)^q$, log-penalties $\psi : t \mapsto \log(|t| + \varepsilon)$, the quantity $\varepsilon > 0$ being here to ensure the function ψ to be differentiable at 0. Other nonconvex regularization functions have been proposed in the statistics community, such as the SCAD penalty [43].

The main motivation for such approaches is that these penalties induce more sparsity than the ℓ_1-norm, but can be addressed with other tools than greedy methods. The unit balls corresponding to the ℓ_q pseudo-norms and norms for several values of q are illustrated in Figure 7.1. When q decreases, the ℓ_q-ball get "closer" to the ℓ_0-ball, and better induces sparsity.



(a) ℓ_0-ball, 2D (b) $\ell_{0.5}$-ball, 2D (c) ℓ_1-ball, 2D (d) ℓ_2-ball, 2D

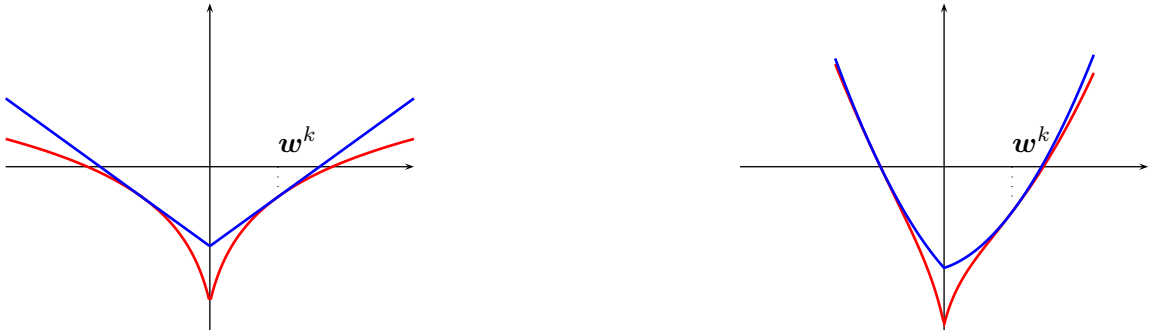
Figure 7.1: Unit balls in 2D corresponding to ℓ_q-penalties.

Even though the optimization problem (7.1) is not convex and not smooth, it is possible to iteratively decrease the value of the objective function by solving a sequence of convex problems. Such algorithmic schemes appear early in the optimization literature under the name DC (difference of convex) programming (see [48] and references therein), and were recently revisited in reweighted-ℓ_1 algorithms [29]. The idea here is relatively simple. It consists at iteration k of the algorithm of minimizing a convex surrogates $\tilde{g}_k$ which is tangent to the graph of the objective function around the current estimate $\boldsymbol{w}^k$.

Obtaining such a surrogate is easy when exploiting the concavity of the functions ψ on $\mathbb{R}^+$, which are always below their tangents, as illustrated in Figure 7.2. The iterative scheme can then be written

$$\boldsymbol{w}^{k+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{w}\in\mathbb{R}^p} f(\boldsymbol{w}) + \lambda \sum_{i=1}^{p} \psi'(|\boldsymbol{w}_i^k|)|\boldsymbol{w}_i|,$$

which is a reweighted-ℓ_1 sparse decomposition problem. To initialize the algorithm, the first step is usually a simple Lasso, with no weights. In practice, the effect of the weights $\psi'(|\boldsymbol{w}_i^k|)$ is to push to zero the smallest non-zero coefficients from iteration $k-1$, and two or three iterations are usually enough to obtain the desired sparsifying effect.



(a) red: $\psi(\boldsymbol{w}) = \log(|\boldsymbol{w}| + \varepsilon)$.
blue: convex surrogate $\psi'(\boldsymbol{w}^k)|\boldsymbol{w}| + C$.

(b) red: $f(\boldsymbol{w}) + \psi(\boldsymbol{w})$.
blue: $f(\boldsymbol{w}) + \psi'(\boldsymbol{w}^k)|\boldsymbol{w}| + C$

Figure 7.2: Surrogate function used in the DC-programming approach.

7.3 Sparse Matrix Factorization and Dictionary Learning

Sparse linear models for regression in statistics and machine learning assume a linear relation $\boldsymbol{y} \approx \boldsymbol{X}\boldsymbol{w}$, where $\boldsymbol{y}$ in $\mathbb{R}^n$ is a vector of observations, $\boldsymbol{X}$ in $\mathbb{R}^{n\times p}$ is a design matrix whose rows can be interpreted as features, and $\boldsymbol{w}$ is a weight vector in $\mathbb{R}^p$. Similar models are used in the signal processing literature, where $\boldsymbol{y}$ is a signal approximated by a linear combination of columns of $\boldsymbol{X}$, which are called dictionary elements, or basis element when $\boldsymbol{X}$ is orthogonal.

Whereas a lot of attention has been devoted to cases where $\boldsymbol{X}$ is fixed and pre-defined, other works have addressed the problem of learning $\boldsymbol{X}$ from training data. In the context of sparse linear models, this problem was first introduced in the neuroscience community by Olshausen and Field [93] to model the spatial receptive fields of simple cells in the mammalian visual cortex. Concretely, given a training set of q signals $\boldsymbol{Y} = [\boldsymbol{y}^1, \ldots, \boldsymbol{y}^q]$ in $\mathbb{R}^{n\times q}$, one looks for a dictionary matrix $\boldsymbol{X}$ in $\mathbb{R}^{n\times p}$ and a coefficient matrix $\boldsymbol{W} = [\boldsymbol{w}^1, \ldots, \boldsymbol{w}^q]$ in $\mathbb{R}^{p\times q}$ such that each signal $\boldsymbol{y}^i$ admits a sparse approximation $\boldsymbol{X}\boldsymbol{w}^i$. In other words, we want to find a dictionary $\boldsymbol{X}$ and a sparse matrix $\boldsymbol{W}$ such that $\boldsymbol{Y} \approx \boldsymbol{X}\boldsymbol{W}$.

A natural formulation is the following non-convex matrix factorization problem:

$$\min_{\boldsymbol{X}\in\mathcal{X},\boldsymbol{W}\in\mathbb{R}^{n\times q}} \frac{1}{q}\sum_{i=1}^{q}\frac{1}{2}\|\boldsymbol{y}^i-\boldsymbol{X}\boldsymbol{w}^i\|_2^2+\lambda\Omega(\boldsymbol{w}^i), \tag{7.2}$$

where Ω is a sparsity-inducing penalty function, and $\mathcal{X}\subseteq\mathbb{R}^{n\times p}$ is a convex set, which is typically the set of matrices whose columns have less than unit ℓ_2-norm. Without any sparse priors (i.e., for $\lambda=0$), then the solution of this factorization problem is obtained through principal component analysis (PCA) (see, e.g., [28] and references therein). However, when $\lambda>0$, the solution of Eq. (7.2) has a different behavior, and may be used as an alternative to PCA for unsupervised learning.

A successful application of this approach is when the vectors $\boldsymbol{y}^i$ are small natural image patches, for example of size $n=10\times 10$ pixels. A typical setting is to have an overcomplete dictionary—that is, the number of dictionary elements can be greater than the signal dimension but a large number of training signals, for example $p=200$ and $q=100\,000$. For this sort of data, dictionary learning finds linear subspaces of small dimension where the patches live, leading to effective applications in image processing [41]. Examples of dictionary elements are given in Figure 7.3.

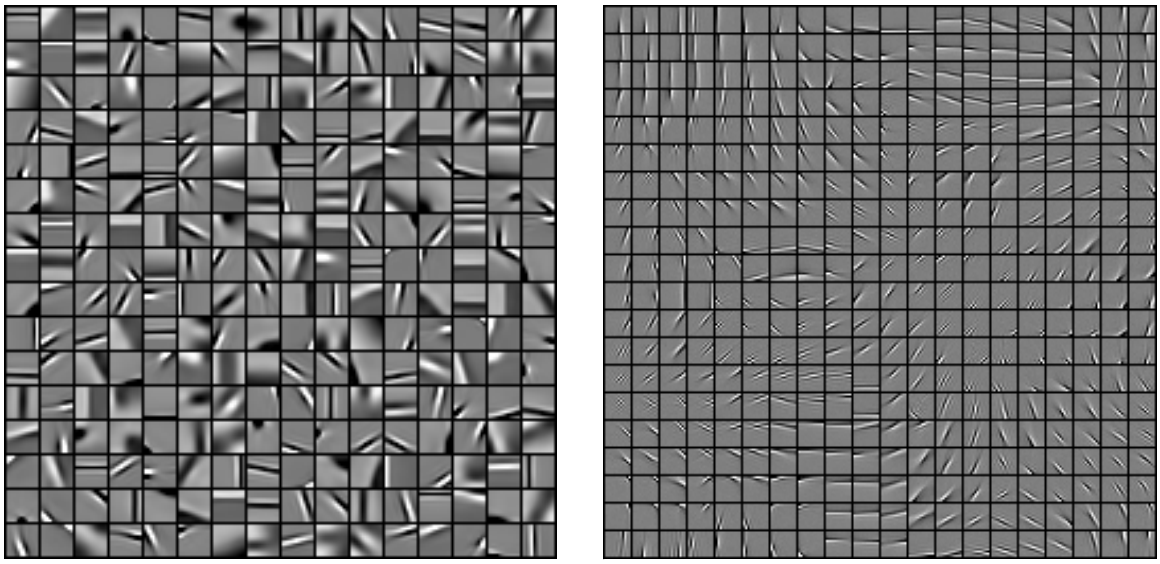


Figure 7.3: Left: Example of dictionary with $p=256$ elements, learned on a database of natural 12×12 image patches when Ω is the ℓ_1-norm. Right: Dictionary with $p=400$ elements, learned with a structured sparsity-inducing penalty Ω (see [77]).

In terms of optimization, Eq. (7.2) is nonconvex and no algorithm has a guarantee of providing a global optimum in general, whatever the choice of penalty Ω is. A typical approach is a block-coordinate scheme, which optimizes $\boldsymbol{X}$ and $\boldsymbol{W}$ in turn, while keeping the other one fixed [42]. Other alternatives include the K-SVD algorithm [3], and online learning techniques [75, 93] that have proven to be particularly efficient when the number

of signals q is large. Convex relaxations of dictionary learning have also been proposed in [14, 26].

7.4 Bayesian Methods

While our survey paper focused mainly on frequentist approaches to sparsity, i.e., approaches that minimize regularized empirical losses, many of the norms that we consider in Section 1.3 may be considered in a Bayesian framework.

For example, the first (naive) Bayesian interpretation of the Lasso is simply a maximum a posteriori (MAP) estimate in a Gaussian linear model, with independent Laplace priors on the loading vectors $\mathbf{w}_i$ (see, e.g., [106]). However, when full Bayesian inference is performed, i.e., the full posterior distribution of $\mathbf{w}$ is considered instead of simply its mode, exact zeros occur with probability zero, while small values are often obtained. In the Bayesian setting, sparse methods are thus often turned into methods with heavy-tail priors (having exact zeros or small values do not change significantly the predictive performance).

The heavy-tailed prior distribution on $\mathbf{w}_i$ is thus key to obtaining posterior estimates with many small values, and this effect is stronger when the tails are heavier, in particular with Student's t-distribution. An important computational (in particular for variational methods) and theoretical aspect of these prior distributions is that they can be expressed as scaled mixture of Gaussians [5, 30], and thus enter the classical framework of automatic relevance determination [85]. Note that in order to perform empirical Bayes estimation (e.g., maximum likelihood estimation of hyperparameters), iterative methods based on DC programming may be efficiently used [125].

Another line of work using Bayesian inference considers using priors on the loading vectors $\mathbf{w}_i$ that put non-zero mass on exact zeros, leading to so-called "spike-and-slab" priors [54]. However, inference with such priors does not lead to convex optimization problems, and sampling methods, while also simple to implement, do not have any guarantees, in particular in high-dimensional settings.

Chapter 8

Quantitative Evaluation

To illustrate and compare the methods presented in this paper, we consider in this section three benchmarks. These benchmarks are chosen to be representative of problems regularized with sparsity-inducing norms, involving different norms and different loss functions. To make comparisons that are as fair as possible, each algorithm is implemented in `C/C++`, using efficient BLAS and LAPACK libraries for basic linear algebra operations. Most of these implementations have been made available in the open-source software SPAMS[1]. All subsequent simulations are run on a single core of a 3.07Ghz CPU, with 8GB of memory. In addition, we take into account several criteria which strongly influence the convergence speed of the algorithms. In particular, we consider

(a) different problem scales,

(b) different levels of correlations between input variables,

(c) different strengths of regularization.

We also show the influence of the required precision by monitoring the time of computation as a function of the objective function.

For the convenience of the reader, we list here the algorithms compared and the acronyms we use to refer to them throughout this section: the homotopy/LARS algorithm (LARS), coordinate-descent (CD), reweighted-ℓ_2 schemes (Re-ℓ_2), simple proximal method (ISTA) and its accelerated version (FISTA). Note that all methods except the working set methods are very simple to implement as each iteration is straightforward (for proximal methods such as FISTA or ISTA, as long as the proximal operator may be computed efficiently). On the contrary, as detailed in Section 6.2, homotopy methods require some care in order to achieve the performance we report in this section.

We will also include in the comparisons generic algorithms such as a subgradient descent algorithm (SG), and a commercial software[2] for cone (CP), quadratic (QP) and second-order cone programming (SOCP) problems.

[1] `http://www.di.ens.fr/willow/SPAMS/`

[2] Mosek, available at `http://www.mosek.com/`.

8.1 Speed Benchmarks for Lasso

We first present a large benchmark evaluating the performance of various optimization methods for solving the Lasso.

We perform small-scale ($n = 200, p = 200$) and medium-scale ($n = 2000, p = 10000$) experiments. We generate design matrices as follows. For the scenario with low correlations, all entries of $\boldsymbol{X}$ are independently drawn from a Gaussian distribution $\mathcal{N}(0, 1/n)$, which is a setting often used to evaluate optimization algorithms in the literature. For the scenario with large correlations, we draw the rows of the matrix $\boldsymbol{X}$ from a multivariate Gaussian distribution for which the *average absolute value* of the correlation between two different columns is eight times the one of the scenario with low correlations. Test data vectors $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w}+\boldsymbol{n}$ where $\boldsymbol{w}$ are randomly generated, with two levels of sparsity to be used with the two different levels of regularization; $\boldsymbol{n}$ is a noise vector whose entries are i.i.d. samples from a Gaussian distribution $\mathcal{N}(0, 0.01\|\boldsymbol{X}\boldsymbol{w}\|_2^2/n)$. In the low regularization setting the sparsity of the vectors $\boldsymbol{w}$ is $s = 0.5\min(n, p)$, and in the high regularization one $s = 0.01\min(n, p)$, corresponding to fairly sparse vectors. For SG, we take the step size to be equal to $a/(k+b)$, where k is the iteration number, and (a, b) are the best[3] parameters selected on a logarithmic grid $(a, b) \in \{10^3, \dots, 10\} \times \{10^2, 10^3, 10^4\}$; we proceeded this way not to disadvantage SG by an arbitrary choice of stepsize.

To sum up, we make a comparison for 8 different conditions (2 scales $\times$ 2 levels of correlation $\times$ 2 levels of regularization). All results are reported on Figures 8.1, 8.2, by averaging 5 runs for each experiment. Interestingly, we observe that the relative performance of the different methods change significantly with the scenario.

Our conclusions for the different methods are as follows:

- **LARS/homotopy methods**: For the small-scale problem, LARS outperforms all other methods for almost every scenario and precision regime. It is therefore *definitely the right choice for the small-scale setting.* Unlike first-order methods, its performance does not depend on the correlation of the design matrix $\boldsymbol{X}$, but rather on the sparsity s of the solution. In our larger scale setting, it has been competitive either when the solution is *very sparse* (high regularization), or when there is *high correlation* in $\boldsymbol{X}$ (in that case, other methods do not perform as well). More importantly, LARS gives an exact solution and computes the regularization path.

- **Proximal methods (ISTA, FISTA)**: FISTA outperforms ISTA in all scenarios but one. Both methods are close for high regularization or low correlation, but FISTA is significantly better for high correlation or/and low regularization. These methods are almost always outperformed by LARS in the small-scale setting, except for *low precision and low correlation.*

  Both methods *suffer from correlated features*, which is consistent with the fact that their convergence rate depends on the correlation between input variables (convergence as a geometric sequence when the correlation matrix is invertible, and as the

[3] "The best step size" is understood here as being the step size leading to the smallest objective function after 500 iterations.

inverse of a degree-two polynomial otherwise). They are *well adapted to large-scale settings, with low or medium correlation.*

- **Coordinate descent (CD)**: The theoretical analysis of these methods suggest that that they behave in a similar way than proximal methods [107, 90]. However, empirically, we have observed that the behavior of CD often translates into a first "warm-up" stage followed by a fast convergence phase.

  Its performance in the *small-scale setting is competitive* (even though always behind LARS), but *less efficient in the large-scale one.* For a reason we cannot explain, *it suffers less than proximal methods from correlated features.*

- **Reweighted-ℓ_2**: This method was outperformed in all our experiments by other dedicated methods.[4] Note that we considered only the smoothed alternating scheme of Section 5 and not first order methods in $\boldsymbol{\eta}$ such as that of [96]. A more exhaustive comparison should include these as well.

- **Generic Methods (SG, QP, CP)**: As expected, generic methods are not adapted for solving the Lasso and are always outperformed by dedicated ones such as LARS.

Among the methods that we have presented, some require an overhead computation of the Gram matrix $\boldsymbol{X}^T\boldsymbol{X}$: this is the case for coordinate descent and reweighted-ℓ_2 methods. We took into account this overhead time in all figures, which explains the behavior of the corresponding convergence curves. Like homotopy methods, these methods could also benefit from an offline pre-computation of $\boldsymbol{X}^T\boldsymbol{X}$ and would therefore be more competitive if the solutions corresponding to several values of the regularization parameter have to be computed.

We have considered in the above experiments the case of the square loss. Obviously, some of the conclusions drawn above would not be valid for other smooth losses. On the one hand, the LARS does no longer apply; on the other hand, proximal methods are clearly still available and coordinate descent schemes, which were dominated by the LARS in our experiments, would most likely turn out to be very good contenders in that setting.

8.2 Group-Sparsity for Multi-task Learning

For ℓ_1-regularized least-squares regression, homotopy methods have appeared in the previous section as one of the best techniques, in almost all the experimental conditions.

This second speed benchmark explores a setting where this homotopy approach cannot be applied anymore. In particular, we consider a multi-class classification problem in the context of cancer diagnosis. We address this problem from a multi-task viewpoint [92]. To this end, we take the regularizer to be ℓ_1/ℓ_2- and ℓ_1/ℓ_∞-norms, with (non-overlapping)

[4]Note that the reweighted-ℓ_2 scheme requires solving iteratively large-scale linear system that are badly conditioned. Our implementation uses LAPACK Cholesky decompositions, but a better performance might be obtained using a pre-conditioned conjugate gradient, especially in the very large scale setting.

groups of variables penalizing features across all classes [92, 70]. As a data-fitting term, we now choose a simple "1-vs-all" logistic loss function.

We focus on two multi-class classification problems in the "small n, large p" setting, based on two datasets[5] of gene expressions. The medium-scale dataset contains $n = 83$ observations, $p = 2308$ variables and 4 classes, while the large-scale one contains $n = 308$ samples, $p = 15009$ variables and 26 classes. Both datasets exhibit highly-correlated features.

In addition to ISTA, FISTA, and SG, we consider here the block coordinate-descent (BCD) from [119] presented in Section 4. We also consider a working-set strategy on top of BCD, that optimizes over the full set of features (including the non-active ones) only one every four iterations. As further discussed in Section 4, it is worth mentioning that the multi-task setting is well suited for [119] since an appropriate approximation of the Hessian can be easily computed.

All the results are reported in Figures 8.3 and 8.4. As expected in the light of the benchmark for the Lasso, BCD appears as the best option, regardless of the sparsity/scale conditions.

8.3 Structured Sparsity

In this second series of experiments, the optimization techniques of the previous sections are further evaluated when applied to other types of loss and sparsity-inducing functions. Instead of the ℓ_1-norm previously studied, we focus on the particular *hierarchical* ℓ_1/ℓ_2-norm Ω introduced in Section 3. From an optimization standpoint, although Ω shares some similarities with the ℓ_1-norm (e.g., the convexity and the non-smoothness), it differs in that it cannot be decomposed into independent parts (because of the overlapping structure of $\mathcal{G}$). CD schemes hinge on this property and as a result, cannot be straightforwardly applied in this case.

8.3.1 Denoising of natural image patches

In this first benchmark, we consider a least-squares regression problem regularized by Ω that arises in the context of the denoising of natural image patches [59]. In particular, based on a hierarchical set of features that accounts for different types of edge orientations and frequencies in natural images, we seek to reconstruct noisy 16×16-patches. Although the problem involves a small number of variables (namely $p = 151$), it has to be solved repeatedly for thousands of patches, at moderate precision. It is therefore crucial to be able to solve this problem efficiently.

The algorithms that take part in the comparisons are ISTA, FISTA, Re-ℓ_2, SG, and SOCP. All results are reported in Figure 8.5, by averaging 5 runs.

We can draw several conclusions from the simulations. First, we observe that across all levels of sparsity, the accelerated proximal scheme performs better, or similarly, than the other

[5]The two datasets we use are *SRBCT* and *14_Tumors*, which are freely available at `http://www.gems-system.org/`.

approaches. In addition, as opposed to FISTA, ISTA seems to suffer in non-sparse scenarios. In the least sparse setting, the reweighted-ℓ_2 scheme matches the performance of FISTA. However this scheme does not yield truly sparse solutions, and would therefore require a subsequent thresholding operation, which can be difficult to motivate in a principled way. As expected, the generic techniques such as SG and SOCP do not compete with the dedicated algorithms.

8.3.2 Multi-class classification of cancer diagnosis.

This benchmark focuses on multi-class classification of cancer diagnosis and reuses the two datasets from the multi-task problem of Section 8.2. Inspired by [63], we build a tree-structured set of groups of features $\mathcal{G}$ by applying Ward's hierarchical clustering [61] on the gene expressions. The norm Ω built that way aims at capturing the hierarchical structure of gene expression networks [63]. For more details about this construction, see [57] in the context of neuroimaging. The resulting datasets with tree-structured sets of features contain $p = 4615$ and $p = 30017$ variables, for respectively the medium- and large-scale datasets.

Instead of the square loss function, we consider the multinomial logistic loss function, which is better suited for multi-class classification problems. As a direct consequence, the algorithms whose applicability crucially depends on the choice of the loss function are removed from the benchmark. This is for instance the case for reweighted-ℓ_2 schemes that have closed-form updates available only with the square loss (see Section 5). Importantly, the choice of the multinomial logistic loss function requires to optimize over a matrix with dimensions p times the number of classes (i.e., a total of $4615 \times 4 \approx 18\,000$ and $30017 \times 26 \approx 780\,000$ variables). Also, for lack of scalability, generic interior point solvers could not be considered here. To summarize, the following comparisons involve ISTA, FISTA, and SG.

All the results are reported in Figure 8.6. The benchmark especially points out that the accelerated proximal scheme performs overall better that the two other methods. Again, it is important to note that both proximal algorithms yield sparse solutions, which is not the case for SG. More generally, this experiment illustrates the flexibility of proximal algorithms with respect to the choice of the loss function.

8.3.3 General overlapping groups of variables

We consider a structured sparse decomposition problem with overlapping groups of ℓ_∞-norms, and compare the proximal gradient algorithm FISTA [17] consider the proximal operator presented in Section 3.3 (referred to as ProxFlow [76]). Since, the norm we use is a sum of several simple terms, we can bring to bear other optimization techniques which are dedicated to this situation, namely proximal splitting method known as alternating direction method of multipliers (ADMM) (see, e.g., [34, 24]). We consider two variants, (ADMM) and (Lin-ADMM)—see more details in [77].

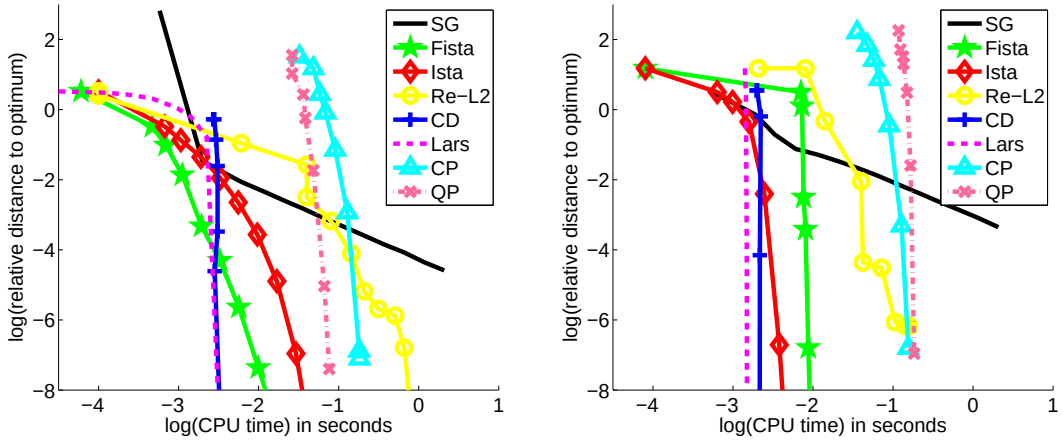
We consider a design matrix $\boldsymbol{X}$ in $\mathbb{R}^{n \times p}$ built from overcomplete dictionaries of discrete cosine transforms (DCT), which are naturally organized on one- or two-dimensional grids

and display local correlations. The following families of groups $\mathcal{G}$ using this spatial information are thus considered: (1) every contiguous sequence of length 3 for the one-dimensional case, and (2) every 3×3-square in the two-dimensional setting. We generate vectors $\boldsymbol{y}$ in $\mathbb{R}^n$ according to the linear model $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w}_0 + \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, 0.01\|\boldsymbol{X}\boldsymbol{w}_0\|_2^2)$. The vector $\boldsymbol{w}_0$ has about 20% nonzero components, randomly selected, while respecting the structure of $\mathcal{G}$, and uniformly generated in $[-1, 1]$.

In our experiments, the regularization parameter λ is chosen to achieve the same level of sparsity (20%). For SG, ADMM and Lin-ADMM, some parameters are optimized to provide the lowest value of the objective function after 1 000 iterations of the respective algorithms. For SG, we take the step size to be equal to $a/(k+b)$, where k is the iteration number, and (a, b) are the pair of parameters selected in $\{10^{-3}, \dots, 10\} \times \{10^2, 10^3, 10^4\}$. The parameter γ for ADMM is selected in $\{10^{-2}, \dots, 10^2\}$. The parameters (γ, δ) for Lin-ADMM are selected in $\{10^{-2}, \dots, 10^2\} \times \{10^{-1}, \dots, 10^8\}$. For interior point methods, since problem (1.1) can be cast either as a quadratic (QP) or as a conic program (CP), we show in Figure 8.7 the results for both formulations. On three problems of different sizes, with $(n, p) \in \{(100, 10^3), (1024, 10^4), (1024, 10^5)\}$, our algorithms ProxFlow, ADMM and Lin-ADMM compare favorably with the other methods, (see Figure 8.7), except for ADMM in the large-scale setting which yields an objective function value similar to that of SG after 10^4 seconds. Among ProxFlow, ADMM and Lin-ADMM, ProxFlow is consistently better than Lin-ADMM, which is itself better than ADMM. Note that for the small scale problem, the performance of ProxFlow and Lin-ADMM is similar. In addition, note that QP, CP, SG, ADMM and Lin-ADMM do not obtain sparse solutions, whereas ProxFlow does.

8.4 General Comments

We conclude this section by a couple of general remarks on the experiments that we presented. First, the use of proximal methods is often advocated because of their optimal worst case complexities in $O(\frac{1}{t^2})$ (where t is the number of iterations). In practice, in our experiments, these and several other methods exhibit empirically convergence rates that are at least linear, if not better, which suggests that the adaptivity of the method (e.g., its ability to take advantage of local curvature) might be more crucial to its practical success. Second, our experiments concentrated on regimes that are of interest for sparse methods in machine learning where typically p is larger than n and where it possible to find good sparse solutions. The setting where n is much larger than p was out of scope here, but would be worth a separate study, and should involve methods from stochastic optimization. Also, even though it might make sense from an optimization viewpoint, we did not consider problems with low levels of sparsity, that is with more dense solution vectors, since it would be a more difficult regime for many of the algorithms that we presented (namely LARS, CD or proximal methods).



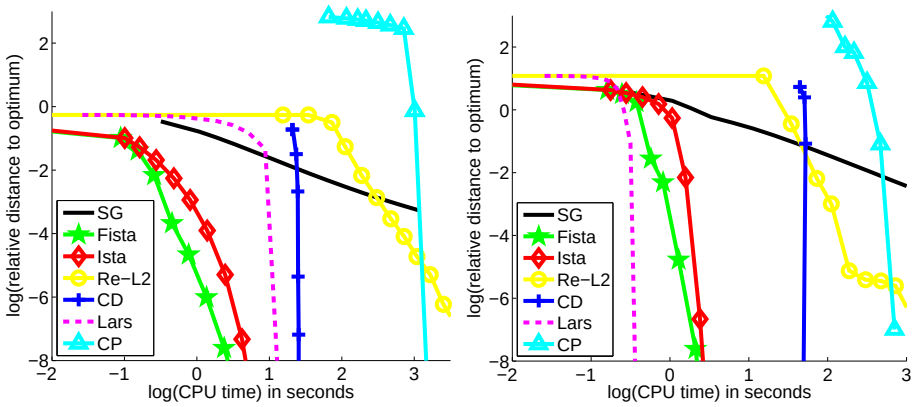
(a) corr: low, reg: low

(b) corr: low, reg: high

(c) corr: high, reg: low

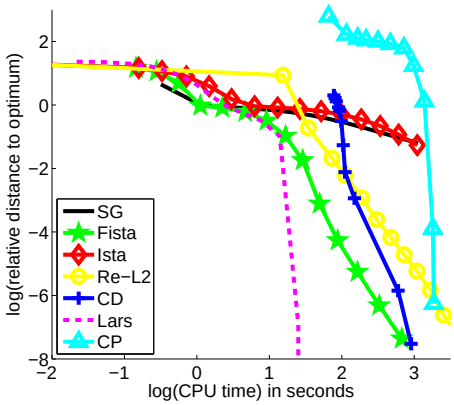
(d) corr: high, reg: high

Figure 8.1: Benchmark for solving the Lasso for the small-scale experiment ($n = 200$, $p = 200$), for the two levels of correlation and two levels of regularization, and 8 optimization methods (see main text for details). The curves represent the relative value of the objective function as a function of the computational time in second on a $\log_{10} / \log_{10}$ scale.

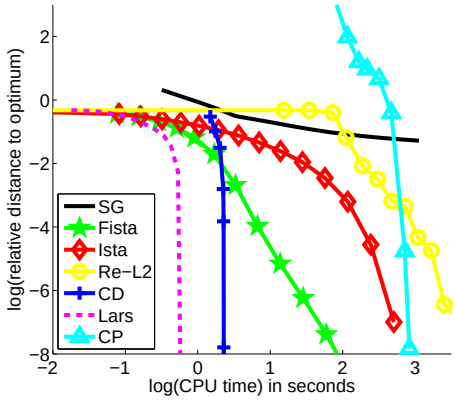


(a) corr: low, reg: low

(b) corr: low, reg: high

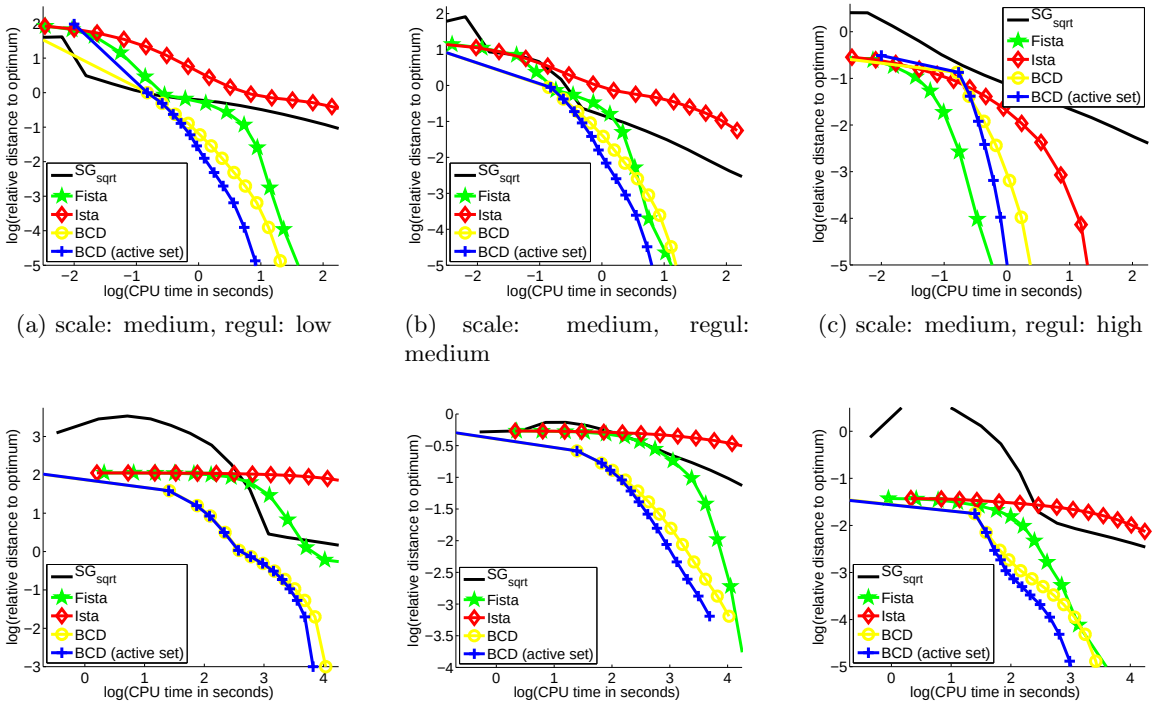


(c) corr: high, reg: low



(d) corr: high, reg: high

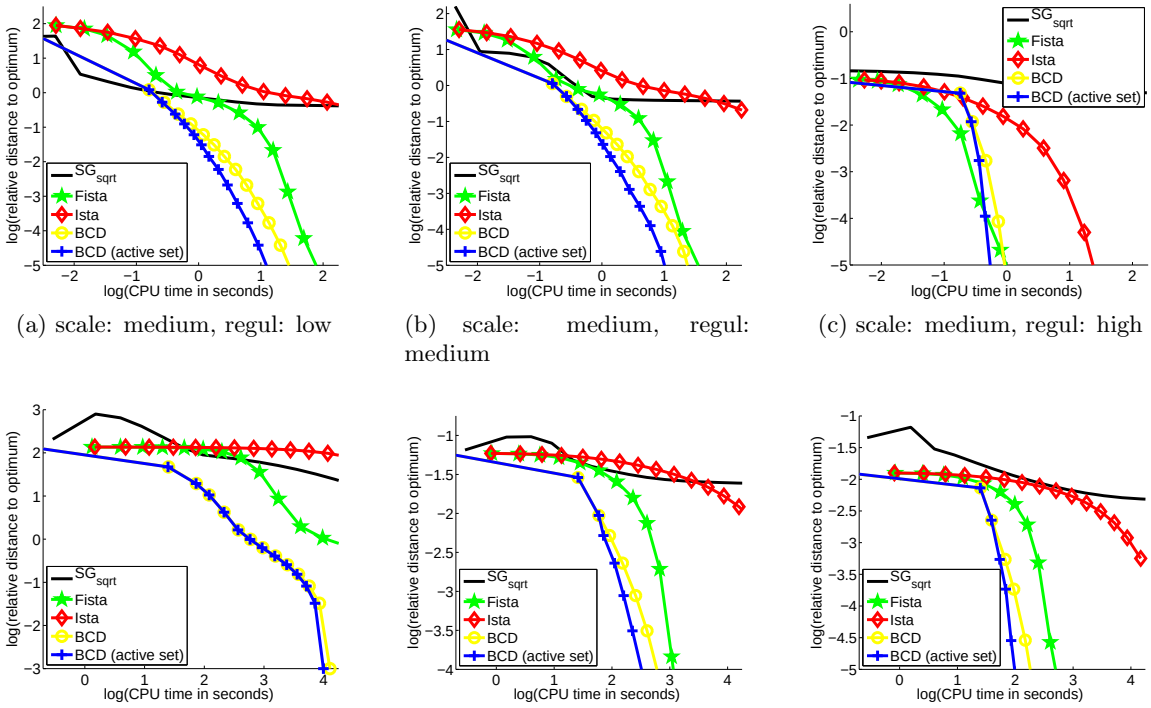
Figure 8.2: Benchmark for solving the Lasso for the medium-scale experiment $n = 2000$, $p = 10000$, for the two levels of correlation and two levels of regularization, and 8 optimization methods (see main text for details). The curves represent the relative value of the objective function as a function of the computational time in second on a $\log_{10} / \log_{10}$ scale.



(a) scale: medium, regul: low (b) scale: medium, regul: medium (c) scale: medium, regul: high

(d) scale: large, regul: low (e) scale: large, regul: medium (f) scale: large, regul: high

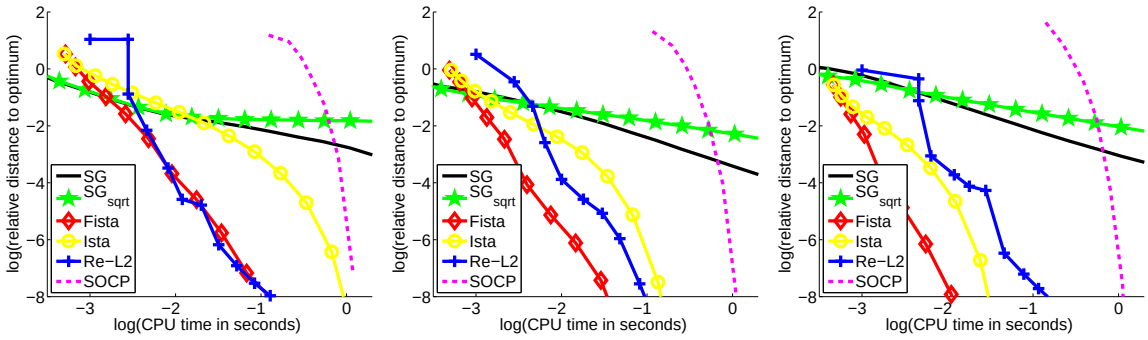
Figure 8.3: Medium- and large-scale multi-class classification problems with an ℓ_1/ℓ_2-regularization, for three optimization methods (see details about the datasets and the methods in the main text). Three levels of regularization are considered. The curves represent the relative value of the objective function as a function of the computation time in second on a $\log_{10}/\log_{10}$ scale. In the highly regularized setting, the tuning of the step-size for the subgradient turned out to be difficult, which explains the behavior of SG in the first iterations.



(a) scale: medium, regul: low (b) scale: medium, regul: medium (c) scale: medium, regul: high

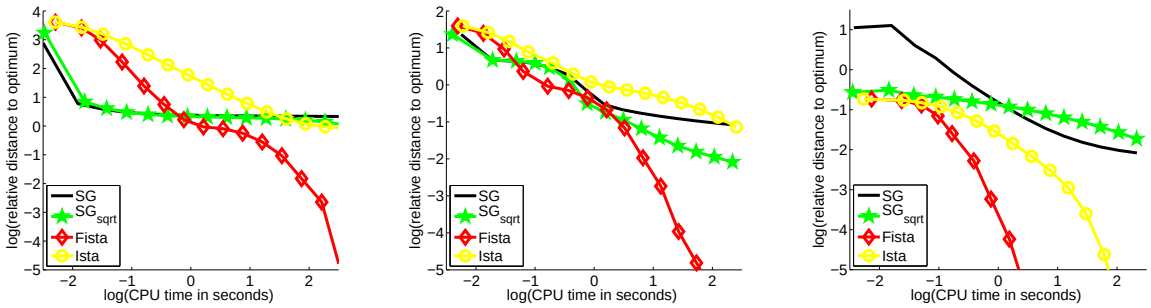
(d) scale: large, regul: low (e) scale: large, regul: medium (f) scale: large, regul: high

Figure 8.4: Medium- and large-scale multi-class classification problems with an ℓ_1/ℓ_∞-regularization for three optimization methods (see details about the datasets and the methods in the main text). Three levels of regularization are considered. The curves represent the relative value of the objective function as a function of the computation time in second on a $\log_{10}/\log_{10}$ scale. In the highly regularized setting, the tuning of the step-size for the subgradient turned out to be difficult, which explains the behavior of SG in the first iterations.

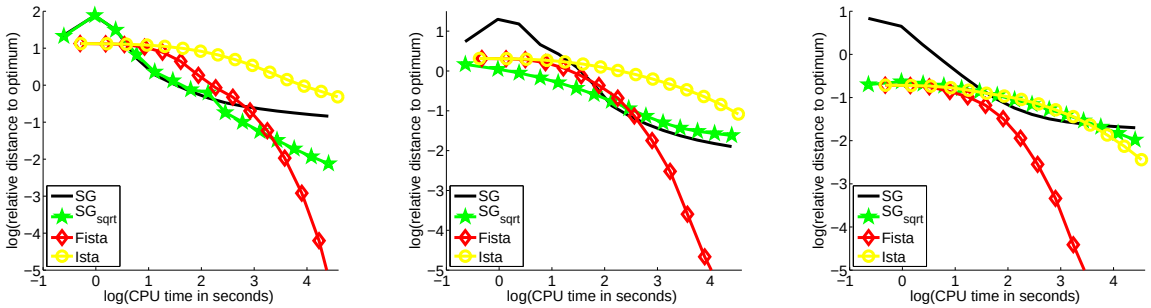


(a) scale: small, regul: low (b) scale: small, regul: medium (c) scale: small, regul: high

Figure 8.5: Benchmark for solving a least-squares regression problem regularized by the hierarchical norm Ω. The experiment is small scale, $n = 256, p = 151$, and shows the performances of five optimization methods (see main text for details) for three levels of regularization. The curves represent the relative value of the objective function as a function of the computational time in second on a $\log_{10} / \log_{10}$ scale.



(a) scale: medium, regul: low (b) scale: medium, regul: medium (c) scale: medium, regul: high



(d) scale: large, regul: low (e) scale: large, regul: medium (f) scale: large, regul: high

Figure 8.6: Medium- and large-scale multi-class classification problems for three optimization methods (see details about the datasets and the methods in the main text). Three levels of regularization are considered. The curves represent the relative value of the objective function as a function of the computation time in second on a $\log_{10} / \log_{10}$ scale. In the highly regularized setting, the tuning of the step-size for the subgradient turned out to be difficult, which explains the behavior of SG in the first iterations.

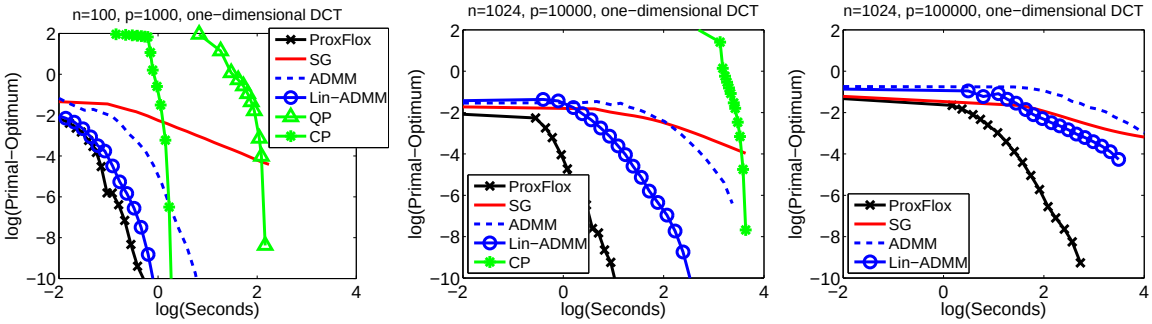


Figure 8.7: Speed comparisons: distance to the optimal primal value versus CPU time (log-log scale). Due to the computational burden, QP and CP could not be run on every problem.

Chapter 9

Extensions

We obviously could not cover exhaustively the literature on algorithms for sparse methods in this chapter.

Surveys and comparisons of algorithms for sparse methods have been proposed by [103] and [130]. These papers present quite a few algorithms, but focus essentially on ℓ_1-regularization and unfortunately do not consider proximal methods. Also, it is not clear that the metrics used to compare the performance of various algorithms is the most relevant to machine learning; in particular, we present the full convergence curves that we believe are more informative than the ordering of algorithms at fixed precision.

Beyond the material presented here, there a few topics that we did not develop and that are worth mentioning.

In terms of algorithms, it is possible to relax the smoothness assumptions that we made on the loss. For instance, some proximal methods are applicable with weaker smoothness assumptions on the function f, such as the Douglas-Rachford algorithm (see details in [34]). The related augmented Lagrangian techniques (see [24, 50, 34] and numerous references therein) or more precisely their variants known as alternating-direction methods of multipliers are also relevant in that setting. These methods are in particular applicable to cases where several regularizations are mixed.

In the context of proximal methods, the metric used to define the proximal operator can be modified by judicious rescaling operations, in order to fit better the geometry of the data [39]. Moreover, they can be mixed with Newton and quasi-Newton methods, for further acceleration (see, e.g., [72]).

Finally, from a broader outlook, our—*a priori* deterministic—optimization problem (1.1) may also be tackled with stochastic optimization approaches, which has been the focus of much recent research [21, 23, 108, 129].

Chapter 10

Conclusions

We presented and compared four families of algorithms for sparse methods: proximal methods, block-coordinate descent algorithms, reweighted-ℓ_2 algorithms and the LARS that are representative of the state of the art. We did not aim at being exhaustive. The properties of these methods can be summarized as follows:

- Proximal methods provide efficient and scalable algorithms that are applicable to a wide family of loss functions, that are simple to implement, compatible with many sparsity-inducing norms and often competitive with the other methods considered.
- For the square loss, the homotopy method remains the fastest algorithm for (a) small and medium scale problems, since its complexity depends essentially on the size of the active sets, (b) cases with very correlated designs. It computes the whole path up to a certain sparsity level. Its main drawback is that it is difficult to implement efficiently, and it is subject to numerical instabilities. On the other hand, coordinate descent and proximal algorithms are trivial to implement.
- For smooth losses, block-coordinate descent provides one of the fastest algorithms but it is limited to separable regularizers.
- For the square-loss and possibly sophisticated sparsity inducing regularizers, ℓ_2-reweighted algorithms provides generic algorithms, that are still pretty competitive compared to subgradient and interior point methods. For general losses, these methods currently require to solve iteratively ℓ_2-regularized problems and it would be desirable to relax this constraint.

Acknowledgements

Francis Bach, Rodolphe Jenatton and Guillaume Obozinski are supported in part by ANR under grant MGA ANR-07-BLAN-0311 and the European Research Council (SIERRA Project). Julien Mairal is supported by the NSF grant SES-0835531 and NSF award CCF-0939370.

Bibliography

[1] J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *Journal of Machine Learning Research*, 10:803–826, 2009.

[2] J. Aflalo, A. Ben-Tal, C. Bhattacharyya, J.S. Nath, and S. Raman. Variable sparsity kernel learning. *The Journal of Machine Learning Research*, 12:565–592, 2011.

[3] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Processing*, 54(11):4311–4322, 2006.

[4] Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the International Conference on Machine Learning*, 2007.

[5] C. Archambeau and F. Bach. Sparse probabilistic projections. In *Advances in Neural Information Processing Systems 21 (NIPS)*, 2008.

[6] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[7] F. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

[8] F. Bach. Consistency of trace norm minimization. *Journal of Machine Learning Research*, 9:1019–1048, 2008.

[9] F. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *Neural Information Processing Systems*, volume 21, 2008.

[10] F. Bach. Structured sparsity-inducing norms through submodular functions. In *Adv. NIPS*, 2010.

[11] F. Bach. Shaping level sets with submodular functions. Technical Report 00542949-v2, HAL, 2011.

[12] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Convex optimization with sparsity-inducing norms. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011.

[13] F. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2004.

[14] F. Bach, J. Mairal, and J. Ponce. Convex sparse matrix factorizations. *Arxiv preprint arXiv:0812.1869*, 2008.

[15] R.G. Baraniuk, V. Cevher, M. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Trans. Information Theory*, 56(4):1982–2001, 2010.

[16] FL Bauer, J. Stoer, and C. Witzgall. Absolute and monotonic norms. *Numerische Mathematik*, 3(1):257–264, 1961.

[17] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[18] D. P. Bertsekas. *Nonlinear programming*. Athena Scientific Belmont, 1999.

[19] P. Bickel, Y. Ritov, and A. Tsybakov. Simultaneous analysis of Lasso and Dantzig selector. *Annals of Statistics*, 37(4):1705–1732, 2009.

[20] J.M. Borwein and A.S. Lewis. *Convex analysis and nonlinear optimization: Theory and Examples*. Springer-Verlag, 2006.

[21] L. Bottou. Online algorithms and stochastic approximations. *Online Learning and Neural Networks*, 5, 1998.

[22] L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Neural Information Processing Systems*, volume 20, pages 161–168, 2008.

[23] L. Bottou and Y. LeCun. Large scale online learning. In *Advances in Neural Information Processing Systems*, volume 16, pages 217–224, 2004.

[24] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–124, 2011. to appear.

[25] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.

[26] D.M. Bradley and J.A. Bagnell. Convex coding. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 83–90, 2009.

[27] P. Brucker. An $O(n)$ algorithm for quadratic knapsack problems. *Operations Research Letters*, 3(3):163–166, 1984.

[28] C. Burges. Dimension reduction: A guided tour. *Machine Learning*, 2(4):275–365, 2009.

[29] E.J. Candès, M. Wakin, and S. Boyd. Enhancing sparsity by reweighted l1 minimization. *Journal of Fourier Analysis and Applications*, 14:877–905, 2008.

[30] F. Caron and A. Doucet. Sparse Bayesian nonparametric regression. In *25th International Conference on Machine Learning (ICML)*, 2008.

[31] V. Cehver, M. Duarte, C. Hedge, and R.G. Baraniuk. Sparse signal recovery using markov random fields. In *Neural Information Processing Systems*, 2008.

[32] A. Chambolle. Total variation minimization and a class of binary MRF models. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 136–152. Springer, 2005.

[33] S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61, 1999.

[34] P.L. Combettes and J.C. Pesquet. *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, chapter Proximal Splitting Methods in Signal Processing. New York: Springer-Verlag, 2010.

[35] P.L. Combettes and V.R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, 4(4):1168–1200, 2006.

[36] S.F. Cotter, J. Adler, B. Rao, and K. Kreutz-Delgado. Forward sequential algorithms for best basis selection. In *IEEE Proceedings of Vision Image and Signal Processing*, pages 235–244, 1999.

[37] I. Daubechies, R. DeVore, M. Fornasier, and C. S. Güntürk. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics*, 63(1):1–38, 2010.

[38] D.L. Donoho and I.M. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association*, 90(432):1200–1224, 1995.

[39] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 2011. to appear.

[40] B. Efron, T. Hastie, and R. Johnstone, I.and Tibshirani. Least angle regression. *Annals of Statistics*, 32(2):407–499, 2004.

[41] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Processing*, 15(12):3736–3745, 2006.

[42] K. Engan, S.O. Aase, H. Husoy, et al. Method of optimal directions for frame design. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 1999. ICASSP'99.*, volume 5, pages 2443–2446, 1999.

[43] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

[44] M. Fazel, H. Hindi, and S. P. Boyd. A rank minimization heuristic with application to minimum order system approximation. In *Proceedings American Control Conference*, volume 6, pages 4734–4739, 2001.

[45] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. *preprint*, 2010.

[46] J.H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, 2001.

[47] W.J Fu. Penalized regressions: the bridge versus the lasso. *Journal of computational and graphical statistics*, 7(3):397–416, 1998.

[48] G. Gasso, A. Rakotomamonjy, and S. Canu. Recovering sparse signals with non-convex penalties and dc programming. *IEEE Trans. Signal Processing*, 57(12):4686–4698, 2009.

[49] A. Genkin, D.D Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

[50] R. Glowinski and P. Le Tallec. *Augmented Lagrangian and operator-splitting methods in nonlinear mechanics*. Society for Industrial Mathematics, 1989.

[51] K.K. Herrity, A.C. Gilbert, and J.A. Tropp. Sparse approximation via iterative thresholding. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, (ICASSP)*, volume 3, 2006.

[52] J. Huang and T. Zhang. The benefit of group sparsity. Technical report, 2009. Preprint arXiv:0901.2962.

[53] J. Huang, Z. Zhang, and D. Metaxas. Learning with structured sparsity. In *Proc. Intl. Conf. Machine Learning*, 2009.

[54] H. Ishwaran and J.S. Rao. Spike and slab variable selection: frequentist and Bayesian strategies. *Annals of Statistics*, 33(2):730–773, 2005.

[55] L. Jacob, G. Obozinski, and J.-P. Vert. Group Lasso with overlaps and graph Lasso. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[56] R. Jenatton, J-Y. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. Technical report, 2009. Preprint arXiv:0904.3523v1.

[57] R. Jenatton, A. Gramfort, V. Michel, G. Obozinski, F. Bach, and B. Thirion. Multi-scale mining of fMRI data with hierarchical structured sparsity. In *International Workshop on Pattern Recognition in Neuroimaging (PRNI)*, 2011.

[58] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. Technical report, Preprint arXiv:1009.2139v2, 2010. To appear in Journal Machine Learning Research.

[59] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010.

[60] R. Jenatton, G. Obozinski, and F. Bach. Structured sparse principal component analysis. In *Proceedings of International Workshop on Artificial Intelligence and Statistics*, 2010.

[61] S.C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[62] K. Kavukcuoglu, M.A. Ranzato, R. Fergus, and Y. Le-Cun. Learning invariant features through topographic filter maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[63] S. Kim and E.P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proc. Intl. Conf. Machine Learning*, 2010.

[64] G.S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *J. Math. Anal. Applicat.*, 33:82–95, 1971.

[65] K. Koh, S. J. Kim, and S. Boyd. An Interior-Point Method for Large-Scale l 1-Regularized Logistic Regression. *Journal of Machine Learning Research*, 8:1555, 2007.

[66] B. Krishnapuram, L. Carin, et al. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 957–968, 2005.

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

[68] G.R.G. Lanckriet, N. Cristianini, L. El Ghaoui, P. Bartlett, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[69] H. Lee, A. Battle, R. Raina, and A.Y. Ng. Efficient sparse coding algorithms. In *Neural Information Processing Systems*, volume 20, 2007.

[70] H. Liu, M. Palatucci, and J. Zhang. Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 649–656, 2009.

[71] K. Lounici, M. Pontil, A. B. Tsybakov, and S. van de Geer. Taking advantage of sparsity in multi-task learning. Technical report, Preprint arXiv:0903.1468, 2009.

[72] S. Sra M. Schmidt, D. Kim. Projected Newton-type methods in machine learning. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011.

[73] N. Maculan and G. Galdino de Paula. A linear-time median-finding algorithm for projecting a vector on the simplex of $\mathbb{R}^n$. *Operations research letters*, 8(4):219–222, 1989.

[74] J. Mairal. *Sparse coding for machine learning, image processing and computer vision*. PhD thesis, Ecole Normale Supérieure de Cachan, 2010. `http://tel.archives-ouvertes.fr/tel-00595312`.

[75] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

[76] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Network flow algorithms for structured sparsity. In *Neural Information Processing Systems*, 2010.

[77] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Convex and network flow optimization for structured sparsity. Technical report, Preprint arXiv:1104.1872, 2011.

[78] S. Mallat and Z. Zhang. Matching pursuit in a time-frequency dictionary. *IEEE Trans. Signal Processing*, 41(12):3397–3415, 1993.

[79] H. Markowitz. Portfolio selection. *Journal of Finance*, 7(1):77–91, 1952.

[80] A.F.T. Martins, N.A. Smith, P.M.Q. Aguiar, and M.A.T. Figueiredo. Structured sparsity in structured prediction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2011.

[81] C.A. Micchelli, J.M. Morales, and M. Pontil. Regularizers for structured sparsity. Technical report, 2011. Preprint arXiv:1010.0556v2.

[82] J.J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *CR Acad. Sci. Paris Sér. A Math*, 255:2897–2899, 1962.

[83] S. Mosci, L. Rosasco, M. Santoro, A. Verri, and S. Villa. Solving structured sparsity regularization with proximal methods. *Machine Learning and Knowledge Discovery in Databases*, pages 418–433, 2010.

[84] B.K. Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24:227, 1995.

[85] R.M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Verlag, 1996.

[86] S. Negahban, P. Ravikumar, M.J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of M-estimators with decomposable regularizers. In *Neural Information Processing Systems*, 2009.

[87] Y. Nesterov. *Introductory lectures on convex optimization: a basic course*. Kluwer Academic Publishers, 2004.

[88] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

[89] Y. Nesterov. Gradient methods for minimizing composite objective function. Technical report, Center for Operations Research and Econometrics (CORE), Catholic University of Louvain, Tech. Rep, 2007.

[90] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *Core discussion papers*, 2010.

[91] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Verlag, 2006. second edition.

[92] G. Obozinski, B. Taskar, and M.I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20(2):231–252, 2009.

[93] B.A. Olshausen and D.J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[94] M.R. Osborne, B. Presnell, and B.A. Turlach. On the Lasso and its dual. *Journal of Computational and Graphical Statistics*, 9(2):319–37, 2000.

[95] M. Pontil, A. Argyriou, and T. Evgeniou. Multi-task feature learning. In *Advances in Neural Information Processing Systems*, 2007.

[96] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

[97] N.S. Rao, R.D. Nowak, S.J. Wright, and N.G. Kingsbury. Convex approaches to model wavelet sparsity patterns. In *International Conference on Image Processing (ICIP)*, 2011.

[98] F. Rapaport, E. Barillot, and J.-P. Vert. Classification of arrayCGH data using fused SVM. *Bioinformatics*, 24(13):i375–i382, Jul 2008.

[99] P. Ravikumar, J. Lafferty, H. Liu, and L. Wasserman. Sparse additive models. *Journal of the Royal Statistical Society. Series B, statistical methodology*, 71:1009–1030, 2009.

[100] K. Ritter. Ein verfahren zur lösung parameterabhängiger, nichtlinearer maximum-probleme. *Mathematical Methods of Operations Research*, 6(4):149–166, 1962.

[101] R.T. Rockafellar. *Convex analysis*. Princeton University Press, 1997.

[102] V. Roth and B. Fischer. The Group-Lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proc. Intl. Conf. Machine Learning*, 2008.

[103] M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for L1 regularization: A comparative study and two new approaches. *Machine Learning: ECML 2007*, pages 286–297, 2007.

[104] M. Schmidt and K. Murphy. Convex structure learning in log-linear models: Beyond pairwise potentials. In *Proceedings of International Workshop on Artificial Intelligence and Statistics*, 2010.

[105] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2001.

[106] M.W. Seeger. Bayesian inference and optimal design for the sparse linear model. *The Journal of Machine Learning Research*, 9:759–813, 2008.

[107] S. Shalev-Shwartz and A. Tewari. Stochastic methods for ℓ_1-regularized loss minimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[108] A. Shapiro, D. Dentcheva, A. Ruszczyński, and A.P. Ruszczyński. *Lectures on stochastic programming: modeling and theory*. Society for Industrial Mathematics, 2009.

[109] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[110] S.K. Shevade and S.S. Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246, 2003.

[111] P. Sprechmann, I. Ramirez, G. Sapiro, and Y.C. Eldar. Collaborative Hierarchical Sparse Modeling. *Arxiv preprint arXiv:1003.0400*, 2010.

[112] N. Srebro, J. D. M. Rennie, and T. S. Jaakkola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems 17*, 2005.

[113] M. Szafranski, Y. Grandvalet, and P. Morizet-Mahoudeaux. Hierarchical penalization. In *Neural Information Processing Systems*, volume 20, 2007.

[114] R. Tibshirani. Regression shrinkage and selection via the Lasso. *J. Roy. Stat. Soc. B*, 58(1):267–288, 1996.

[115] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused Lasso. *J. Roy. Stat. Soc. B*, 67(1):91–108, 2005.

[116] J.A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Signal Processing*, 50(10):2231–2242, October 2004.

[117] J.A. Tropp, A.C. Gilbert, and M.J. Strauss. Algorithms for simultaneous sparse approximation. part i: Greedy pursuit. *Signal Processing, special issue "sparse approximations in signal and image processing"*, 86:572–588, 2006.

[118] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. *submitted to SIAM Journal on Optimization*, 2008.

[119] P. Tseng and S. Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117(1):387–423, 2009.

[120] B.A. Turlach, W.N. Venables, and S.J. Wright. Simultaneous variable selection. *Technometrics*, 47(3):349–363, 2005.

[121] G. Varoquaux, R. Jenatton, A. Gramfort, G. Obozinski, B. Thirion, and F. Bach. Sparse structured dictionary learning for brain resting-state activity modeling. In *NIPS Workshop on Practical Applications of Sparse Modeling: Open Issues and New Directions*, 2010.

[122] J.P. Vert and K. Bleakley. Fast detection of multiple change-points shared by many signals using group LARS. In *Neural Information Processing Systems*, volume 23, 2010.

[123] M. J. Wainwright. Sharp thresholds for noisy and high-dimensional recovery of sparsity using ℓ_1- constrained quadratic programming. *IEEE transactions on information theory*, 55(5):2183, 2009.

[124] S. Weisberg. *Applied Linear Regression*. Wiley, 1980.

[125] D. Wipf and S. Nagarajan. A new view of automatic relevance determination. *Advances in neural information processing systems*, 20:1625–1632, 2008.

[126] S.J. Wright. Accelerated block-coordinate relaxation for regularized optimization. Technical report, Technical report, University of Wisconsin-Madison, 2010.

[127] S.J. Wright, R.D. Nowak, and M.A.T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

[128] T.T Wu and K. Lange. Coordinate descent algorithms for lasso penalized regression. *Annals of Applied Statistics*, 2(1):224–244, 2008.

[129] L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 9:2543–2596, 2010.

[130] G.X. Yuan, K.W. Chang, C.J. Hsieh, and C.J. Lin. A comparison of optimization methods for large-scale l1-regularized linear classification. Technical report, Department of Computer Science, National University of Taiwan, 2010.

[131] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *J. Roy. Stat. Soc. B*, 68:49–67, 2006.

[132] P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *Annals of Statistics*, 37(6A):3468–3497, 2009.

[133] P. Zhao and B. Yu. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7:2541–2563, 2006.

[134] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B*, 67(2):301–320, 2005.