Proximal methods for the latent group lasso penalty

Silvia Villa⋆, Lorenzo Rosasco†, Sofia Mosci‡, Alessandro Verri‡
⋆ *Istituto Italiano di Tecnologia, Genova, ITALY*
† *CBCL, McGovern Institute, Artificial Intelligence Lab, BCS, MIT, USA*
‡ *DIBRIS, Università di Genova, ITALY*
silvia.villa@iit.it, lrosasco@mit.edu, {sofia.mosci,alessandro.verri}@unige.it,

September 4, 2012

Abstract

We consider a regularized least squares problem, with regularization by structured sparsity-inducing norms, which extend the usual ℓ_1 and the group lasso penalty, by allowing the subsets to overlap. Such regularizations lead to nonsmooth problems that are difficult to optimize, and we propose in this paper a suitable version of an accelerated proximal method to solve them. We prove convergence of a nested procedure, obtained composing an accelerated proximal method with an inner algorithm for computing the proximity operator. By exploiting the geometrical properties of the penalty, we devise a new active set strategy, thanks to which the inner iteration is relatively fast, thus guaranteeing good computational performances of the overall algorithm. Our approach allows to deal with high dimensional problems without pre-processing for dimensionality reduction, leading to better computational and prediction performances with respect to the state-of-the art methods, as shown empirically both on toy and real data.

keywords: Structured sparsity, proximal methods, regularization

AMS Classification: 65K10, 90C25

1 Introduction

Sparsity has become a popular way to deal with a number of problems arising in signal and image processing, statistics and machine learning [18]. In a broad sense, it refers to the possibility of writing the solution in terms of a few building blocks. Often sparsity based methods are the key towards finding interpretable models in real-world problems. For example, sparse regularization based with ℓ_1-type penalties is a powerful approach to find sparse solutions by minimizing a convex functional [47, 11, 17]. The success of ℓ_1 regularization motivated exploring different kinds of sparsity properties for regularized optimization problems, exploiting available a priori information, which restricts the admissible sparsity patterns of the solution. An example of a sparsity pattern is when the variables are partitioned into groups (known a priori), and the goal is to estimate a sparse model where variables belonging to the same group are either jointly selected or discarded. This problem can be solved by regularizing with the group ℓ_1 penalty, also known as group lasso penalty [51]. The latter is the sum, over the groups, of the euclidean norms of the coefficients restricted to each group. Note that, for any $p > 1$, the same groupwise selection can be achieved by regularizing with the ℓ_1/ℓ_p norm, i.e. the sum over the groups of the ℓ_p norm of the coefficients restricted to each group. A possible generalization of the group lasso penalty is obtained considering groups of variables which can be potentially overlapping [52, 23], and the goal is to estimate a model which support is the union of groups. For example, this is a common situation in bioinformatics (especially in the context of high-throughput data such as gene expression and mass spectrometry data), where problems are characterized by a very low number of samples with several thousands of variables. In fact, when the number of

samples is not sufficient to guarantee accurate model estimation, a possible solution is to take advantage of the huge amount of prior knowledge encoded in online databases such as the Gene Ontology [14]. Largely motivated by applications in bioinformatics, the *latent group lasso with overlap penalty* is proposed in [21] and further studied in [35, 2] and in [37] in the image processing context, which generalizes the ℓ_1/ℓ_2 penalty to overlapping groups, thus satisfying the assumption that the admissible sparsity patterns must be unions of a subset of the groups.

All the methods proposed in the literature solve the minimization problem arising in [21] by applying state-of-the-art techniques for group lasso in an expanded space, called *space of latent variables*, built by duplicating variables that belong to more than one group. The most popular optimization strategies that have been proposed are interior-points methods [3, 36], block coordinate descent [27], proximal methods [42, 30, 37, 25, 12] and the related alternating direction method [15]. Very recently, the paper [39] proposed an accelerated alternating direction method and [40] studied a block coordinate descent, along with a proximal method with variable step-sizes.

As already noted in [21], though very natural, every implementation developed in the latent variables does not scale to large datasets: when the groups have significant overlap, a more scalable algorithm with no data duplication is needed. For this reason we propose an alternative optimization approach to solve the group lasso problem with overlap, and extend it to the entire family of group lasso with overlap penalties, that generalize the ℓ_1/ℓ_p penalties to overlapping groups for $p > 1$. Our method is a two-loops iterative scheme based on proximal methods (see for example [32, 6, 5]), and more precisely on the accelerated version named FISTA [5]. It does not require explicit replication of the variables and is thus more appropriate to deal with high dimensional problems with large group overlap. In fact, the proximity operator can be efficiently computed by exploiting the geometrical properties of the penalty. We show that such an operator can be written as the identity minus the projection onto a suitable convex set, which is the intersection of as many convex sets as the number of *active groups*, that is groups corresponding to active constraints, which can be easily found. Indeed, the identification of the active groups is a key step, since it allows computing the projection in a reduced space. For general p, the projection can be solved via the Cyclic Projections algorithm [4]. Furthermore, for the case $p = 2$, we present an accelerated scheme, where the reduced projection is computed by solving a corresponding dual problem via the projected Newton method [7], thus working in a much lower dimensional space.

The present paper completes and extends the preliminary results presented in the short conference version [31]. In particular, it contains a general mathematical presentation and all the proofs, which were omitted in [31]. We next describe how the rest of the paper is organized, and then highlight the main novelties with respect to the short version. In Section 2, we cast the problem of Group-wise Selection with Overlap (GSO) as a regularization problem based on a modified ℓ_1/ℓ_p-type penalty and compare it with other structured sparsity penalties. We extend the approach in [31] for $p = 2$ to general $p > 1$. In Section 3, we describe the derivation of the proposed optimization scheme, and prove its convergence. Precisely, we first recall proximal methods in Subsection 3.1, then in Subsection 3.2 we describe the technical results that ease the computation of the proximity operator as a simplified projection, and present different projection algorithms depending on p. With respect to [31], we show that our active set strategy can be profitably used in this generalized framework in combination with any algorithm chosen to compute the inner projection. Furthermore, to solve the projection for a general $p \in (1, +\infty]$, we discuss the use of a cyclic projections algorithm, whose convergence in norm is guaranteed and results in a rate of convergence for the proposed proximal method, proved in Subsection 3.3. Section 4 is a substantial extension of the experiments performed in [31]. We empirically analyze the computational performance of our optimization procedure. We first study the performance of the different variations of the proposed optimization scheme. Then we present a set of numerical experiments comparing running time of our algorithm with state-of-the-art techniques. We conclude with a real data experiment where we show that the improved computational performance allows dealing with large data sets without preprocessing thus improving also the prediction and selection performance. Finally, in Appendix B we review the projected Newton method [7].

Notation. Given a vector $x \in \mathbb{R}^d$, we denote with $\|\cdot\|_p$ the ℓ_p-norm of x, defined as $\|x\|_p = (\sum_{j=1}^d x_j^p)^{1/p}$ and $\|x\|_\infty = \max_{j\in\{1,\dots d\}} |x_j|$. We will also use the notation $\|x\|_{G,p} = (\sum_{j\in G} x_j^p)^{1/p}$ for $p \geq 1$, and $\|x\|_{G,\infty} = \max_{j\in G} |x_j|$ to denote the ℓ_p-norm of the components of x in $G \subset \{1, \dots, d\}$. When the subscript p is omitted, the ℓ_2 norm is used, $\|\cdot\| = \|\cdot\|_2$. The conjugate exponent of p is denoted by q; we recall that q is such that $1/p + 1/q = 1$. In the following, X will denote $\mathbb{R}^d$ and Y a bounded interval in $\mathbb{R}$.

2 Group-wise selection with Overlap (GSO)

This paper proposes an optimization algorithm for a regularized least-squares problem of the type

$$\min_{x\in\mathbb{R}^d} \mathcal{E}_\tau^p(x), \qquad \mathcal{E}_\tau^p(x) = \frac{1}{n}\left\|\Psi x - y\right\|^2 + 2\tau\Omega_p^{\mathcal{G}}(x)\,, \tag{GSO-p}$$

where $\Psi : \mathbb{R}^d \to \mathbb{R}^n$ is a linear operator, $y \in \mathbb{R}^n$, and $\Omega_p^{\mathcal{G}} : \mathbb{R}^d \to [0, +\infty)$ is a convex and lower semicontinuous penalty, depending on a parameter $p \in (1, +\infty]$, and on an a priori given group structure, $\mathcal{G} = \{G_r\}_{r=1}^B$, with $G_r \subset \{1, \dots, d\}$ and $\bigcup_{r=1}^B G_r = \{1, \dots, d\}$. Note that other data fit terms could be used, different from the quadratic one, as long as they are convex and continuously differentiable with Lipschitz continuous gradient. We will focus on least squares to simplify the exposition. Most group sparsity penalties can be built starting from the family of canonical linear projections on the subspace identified by the indices belonging to G_r, i.e. $P_r : \mathbb{R}^d \to \mathbb{R}^{G_r}$. The definition of the penalties we consider is based on the adjoint of the linear operator

$$P : \mathbb{R}^d \to \prod_{r=1}^B \mathbb{R}^{G_r}, \quad Px = (P_1 x, \dots, P_B x),$$

that is the operator

$$P^* : \prod_{r=1}^B \mathbb{R}^{G_r} \to \mathbb{R}^d, \quad P^*(v_1, \dots, v_B) = \sum_{r=1}^B P_r^* v_r,$$

where $P_r^* : \mathbb{R}^{G_r} \to \mathbb{R}^d$ is the canonical injection. For $x \in \mathbb{R}^d$ we set

$$\Omega_p^{\mathcal{G}}(x) = \min_{\substack{v \in \prod \mathbb{R}^{G_r} \\ P^* v = x}} \sum_{r=1}^B \left\|v_r\right\|_p . \tag{1}$$

For $p = 2$, the functional $\Omega_2^{\mathcal{G}}$ was introduced in [21] (see also [35, 2]). The distinctive feature of the family of penalties $\Omega_p^{\mathcal{G}}$, is that they have the property of inducing group-wise selection, that is they lead to solutions with support (i.e. set of non zero entries) which is the *union* of a subsets of the groups defined a priori. In fact, $\Omega_p^{\mathcal{G}}$ can be seen as a generalization of the mixed ℓ_1/ℓ_p norms, originally introduced for disjoint groups:

$$\mathrm{R}_p^{\mathcal{G}}(x) = \sum_{r=1}^B \left\|x\right\|_{G_r,p}\,, \qquad p \geq 1.$$

For $p = 2$, $\mathrm{R}_p^{\mathcal{G}}$ is the group lasso penalty, and it is well-known [51] that such penalties lead to solutions whose support is the union of a small number of groups. The penalty $\mathrm{R}_p^{\mathcal{G}}$ can be written also if the groups overlap, and more generally the composite absolute penalties (CAP)

$$J_{\gamma,p}^{\mathcal{G}}(x) = \sum_{r=1}^B (\left\|x\right\|_{G_r,p})^\gamma\,,$$

first introduced in [52] and coinciding with $\mathrm{R}_p^{\mathcal{G}}$ for $\gamma = 1$, have been intensively studied. The $J_{\gamma,p}$ penalties allow to deal with complex groups structures involving hierarchies or graphs and it is proved in [23] that the CAP penalties constraint the support to be the *complement of a union* of groups. $\Omega_p^{\mathcal{G}}$ and $\mathrm{R}_p^{\mathcal{G}}$ are thus somehow complementary and have different domain of applications [23, 26, 24].

While many algorithms have been proposed to solve the optimization problem corresponding to $\mathrm{R}_p^{\mathcal{G}}$, the one corresponding $\Omega_p^{\mathcal{G}}$ is much less studied. This is due on the one hand to the fact that the penalty is more complex, and on the other hand to the widespread use of the "replication strategy". The latter is based on the observation that, using the definition of $\Omega_p^{\mathcal{G}}$, and the surjectivity of P^*, the (GSO-p) minimization problem can be written as

$$\min_{v \in \prod_{r=1}^B \mathbb{R}^{G_r}} \frac{1}{n}\left\|\Psi P^* v - y\right\|^2 + 2\tau \sum_{r=1}^B \left\|v_r\right\|_p , \tag{2}$$

which is a group lasso problem without overlap for the linear operator ΨP^* in the so called *latent variables* $(v_r)_{r=1}^B$, obtained by replicating variables belonging to more than one group. The last rewriting allows to apply every algorithm developed for the standard group-lasso to the overlapping case, but this strategy is not feasible for high dimensional problems with large group overlaps, as potentially many artificial dimensions are created. The main goal of this paper is to propose and study an optimization algorithm which does not require the replication of variables belonging to more than one group.

The choice $p > 1$ has both technical and practical motivations. On the one hand, it guarantees convexity of the penalty – which can be shown to be a norm (see Lemma 1 in [21] for $p = 2$) –, and, as a consequence, of the (GSO-p) regularization problem (note that this is valid for $p = 1$ too). On the other hand, it enforces *"democracy"* among the elements that belong to the same group, in the sense that no intragroup sparsity is enforced, thus inducing group-wise selection. The case $p = 1$ is trivial, since the penalty $\Omega_1^{\mathcal{G}}$ coincides with the ℓ_1 norm, or lasso penalty [47]:

$$\Omega_1^{\mathcal{G}}(x) = \inf_{\substack{(v_1,\dots,v_B)\in\prod \mathbb{R}^{G_r} \\ P^*v=x}} \sum_{r=1}^{B}\sum_{j\in G_r}|(v_r)_j| = \inf_{\substack{(v_1,\dots,v_B)\in\prod \mathbb{R}^{G_r} \\ P^*v=x}} \sum_{j=1}^{d}\sum_{r:j\in G_r}^{B}|(v_r)_j| = \sum_{j=1}^{d}|x_j|,$$

and is thus independent of $\mathcal{G}$.

Example 1. A particular instance of the above problem occurs in statistical learning. Assume that the estimator and the regression function can be described by a generalized linear model $f(x) = \sum_{j=1}^d x_j\psi_j(x)$, for a given dictionary $\{\psi_j\}_{j=1}^d$ of functions $\psi_j : X \to Y$ (with X a set and $Y \subseteq \mathbb{R}$). Given a training set $\{(x_i, y_i)_{i=1}^n\} \in (X \times Y)^n$ the regularized empirical risk takes the form

$$\frac{1}{n}\|\Psi x - y\|^2 + 2\tau\Omega_p^{\mathcal{G}}(x),$$

with $\Psi : \mathbb{R}^d \to \mathbb{R}^n$, $\Psi x = \sum_{j=1}^d \psi_j(x_i)x_j$ and $y = (y_1, \dots, y_n)$.

Example 2. Most results obtained in the paper hold in an infinite dimensional setting. In particular, our approach can be naturally extended to the *multiple kernel learning*(MKL) problem [28]. For this problem, given reproducing kernel Hilbert spaces $\mathcal{H}_1, \dots, \mathcal{H}_m$ of functions $g : \mathcal{X} \to \mathbb{R}$, defining $\mathcal{H} = \sum_{r=1}^m \mathcal{H}_r$, the resulting optimization problem takes the form (see [28])

$$\min_{g\in\prod_r \mathcal{H}_r} \left\|\Psi(\sum_r g_r) - y\right\|^2 + \sum_{r=1}^m \|g_r\|_{\mathcal{H}_r},$$

for a suitable $\Psi : \mathcal{H} \to \mathbb{R}^n$, $y \in \mathbb{R}^n$. As can be readily seen, the multiple kernel learning problem has the same structure of the (GSO-p) problem described above.

3 An efficient proximal algorithm

Due to non-smoothness of the penalty term, solving the (GSO-p) minimization problem is not trivial. Moreover, if one needs to solve the (GSO-p) problem for high dimensional data, the use of standard second-order methods such as interior-point methods is precluded (see for instance [6]), since they need to solve large systems of linear equations to compute the Newton steps. On the other hand, first order methods inspired to Nesterov's seminal paper [33] (see also [32]) and based on the proximal map are accurate, and robust, in the sense that their performance does not depend on the fine tuning of various controlling parameters. Furthermore, these methods were already proved to be a computationally efficient alternative for solving many regularized inverse problems in image processing [10], compressed sensing [6] and machine learning applications [2, 16, 30].

3.1 Proximal methods

The (GSO-p) regularized convex functional is the sum of a convex smooth term, $F(x) = \frac{1}{n}\|\Psi x - y\|^2$, with Lipschitz continuous gradient, and a non-differentiable penalty $\tau\Omega_p^{\mathcal{G}}(\cdot)$. A minimizing sequence can be computed with a proximal gradient algorithm [48] (a.k.a. forward-backward splitting method [13], and Iterative Shrinkage Thresholding Algorithm (ISTA) [5])

$$x^m = \text{prox}_{\frac{\tau}{\sigma}\Omega_p^{\mathcal{G}}}\left(x^{m-1} - \frac{1}{2\sigma}\nabla F(x^{m-1})\right) \tag{ISTA}$$

for a suitable choice of σ, and any initialization x^0. Recently, several accelerations of ISTA have been proposed [34, 48, 5]. With respect to ISTA, they only require the additional computation of a linear combination of two consecutive iterates. Among them, FISTA (Fast Iterative Shrinkage Thresholding Algorithm) [5] is given by the following updating rule for $m \geq 1$

$$\begin{aligned} x^m &= \text{prox}_{\frac{\tau}{\sigma}\Omega_p^{\mathcal{G}}}\left(h^m - \frac{1}{2\sigma}\nabla F(h^m)\right) \\ s_{m+1} &= \frac{1}{2}\left(1 + \sqrt{1 + 4s_m^2}\right) \\ h^{m+1} &= \left(1 + \frac{s_m - 1}{s_{m+1}}\right)x^m + \frac{1 - s_m}{s_{m+1}}x^{m-1} \end{aligned} \tag{FISTA}$$

for a suitable choice of $\sigma > 0$, $s_1 = 1$, and any initialization $h^1 = x^0$. Both schemes are based on the computation of the proximity operator [29], which is defined as

$$\text{prox}_{\lambda\Omega_p^{\mathcal{G}}}(z) = \underset{x\in\mathbb{R}^d}{\text{argmin}}\, \Phi_\lambda(x), \qquad \text{with} \quad \Phi_\lambda(x) = \frac{1}{2\lambda}\|x - z\|^2 + \Omega_p^{\mathcal{G}}(x), \qquad \lambda > 0. \tag{3}$$

Algorithm 1 FISTA for GSO-p

Given: $\mathcal{G},\ p \in (1, +\infty],\ \tau > 0,\ \epsilon_0 > 0, \alpha > 0,\ x^0 = h^0 \in \mathbb{R}^d, s_0 = 1,$
Let: $\sigma = ||\Psi^T\Psi||/n, m = 0$ and q such that $\frac{1}{p} + \frac{1}{q} = 1$.
while `convergence not reached` **do**

- $\hat{h}^m = h^m - \frac{1}{n\sigma}\Psi^T(\Psi h^m - y)$
- Find $\hat{\mathcal{G}}^m = \{G \in \mathcal{G}, \left\|\hat{h}^m\right\|_G \geq \frac{\tau}{\sigma}\}$
- Approximately compute the projection of $\hat{h}^m$ onto $\frac{\tau}{\sigma}K_p^{\hat{\mathcal{G}}^m} := \bigcap_{G\in\hat{\mathcal{G}}^m}\left\{h \in \mathbb{R}^d\ :\ \|h\|_{G,q} \leq \frac{\tau}{\sigma}\right\}$ with tolerance $\epsilon_0 m^{-\alpha}$
- $x^m = \hat{h}^m - \pi_{\frac{\tau}{\sigma}K_p^{\hat{\mathcal{G}}^m}}(\hat{h}^m)$
- $s_{m+1} = \frac{1}{2}\left(1 + \sqrt{1 + 4s_m^2}\right)$
- $h^{m+1} = \left(1 + \frac{s_m - 1}{s_{m+1}}\right)x^m + \frac{1-s_m}{s_{m+1}}x^{m-1}$

end while
return x^m

The convergence rate of $\mathcal{E}_\tau^p(x^m) - \min\mathcal{E}_\tau^p$, for ISTA and FISTA, is $O(1/m)$ and $O(1/m^2)$, respectively, when the proximity operator is computed exactly. However, in general, the exact expression is not available. Recently, it has been shown that, also in the presence of errors, the accelerated version maintains advantages with respect to

the basic one. In fact, the rate $O(1/m^2)$ for FISTA in the presence of computational errors was recently proved in [45, 50] for various error criteria. Convergence of ISTA with errors was already known, and first proved in [41, 13].

Since the proximity operator of the penalty $\Omega_p^{\mathcal{G}}$ is not admissible in closed form, the (GSO-p) minimization problem can thus be solved via an inexact version of the iterative schemes ISTA or FISTA, where $\nabla F(h^m)$ is simply $2\Psi^T(\Psi h^m - y)/n$. Note that, in the special case of not overlapping groups, the proximity operator can be explicitly evaluated group-wise, and reduces to a group-wise soft-thresholding operator. In the general case, as explained in Subsection 3.2, the proximity operator can be written in terms of a projection, and we will provide an algorithm to approximately compute it. Note also that we will show that at each step the projection involves only a subset of the initial groups, the active groups, thus significantly increasing the computational performance of the overall algorithm.

3.2 Computing the proximity operator of $\Omega_p^{\mathcal{G}}$

In this subsection we state the lemmas that allow us to efficiently compute the proximity operator of $\Omega_p^{\mathcal{G}}$ and to formulate the inexact version of FISTA reported in Algorithm 1.

As a direct consequence of standard results of convex analysis, Lemma 1 shows that the computation of the proximity operator amounts to the computation of a projection operator onto the intersection of convex sets, each of them corresponding to a group. In Lemma 2, we theoretically justifies an active set strategy, by showing that when projecting a vector onto this intersection, it is possible to discard the constraints which are already satisfied.

Lemma 1. *For any $\lambda > 0$ and $p \geq 1$, the proximity operator of $\lambda\Omega_p^{\mathcal{G}}$, where $\Omega_p^{\mathcal{G}}$ is defined in* (1)*, is given by*

$$prox_{\lambda\Omega_p^{\mathcal{G}}} = I - \pi_{\lambda K_p^{\mathcal{G}}}.$$

where $\pi_{\lambda K_p^{\mathcal{G}}}$ denotes the projection onto $\lambda K_p^{\mathcal{G}}$, and $K_p^{\mathcal{G}}$ is given by

$$K_p^{\mathcal{G}} = \{x \in \mathbb{R}^d, \|x\|_{G_r,q} \leq 1, \textit{ for } r = 1, \ldots, B\}. \tag{4}$$

The proof exploits the particular definition of the penalty and relies on the Moreau decomposition

$$\text{prox}_{\lambda\Omega}(x) = x - \lambda \text{prox}_{\frac{\Omega^*}{\lambda}}\left(\frac{x}{\lambda}\right). \tag{5}$$

Formula (4) allows to compute the proximity operator of Ω starting from the proximity operator of the Fenchel conjugate. In our case, being $\Omega_p^{\mathcal{G}}$ one homogeneous, we obtain the identity minus the projection onto a closed and convex set. The particular geometry of $K_p^{\mathcal{G}}$, which is the intersection of B convex generalized cylinders "centered" on a coordinate subspace, derives from definition of $\Omega_p^{\mathcal{G}}$ and the explicit computation of its Fenchel conjugate. Observe that by definition $\Omega_p^{\mathcal{G}}$ is the infimal convolution of B functions, and precisely the B norms on $\mathbb{R}^{G_r}$ composed with the projections. By standard properties of the Fenchel conjugate, it follows that $(\Omega_p^{\mathcal{G}})^* = \sum \iota_q$, where ι_q is the dual function of $\|\cdot\|_p$, i.e. the indicator function of the ℓ_q unitary ball in $\mathbb{R}^{G_r}$. We give here a self-contained proof which does not use the notion of infimal convolution. A different proof for the case $p = 2$ is given in [35].

Proof. We start by computing explicitly the Fenchel conjugate of $\Omega_p^{\mathcal{G}}$. By definition,

$$\begin{aligned}
(\Omega_p^{\mathcal{G}})^*(u) &= \sup_{x\in\mathbb{R}^d}\left[\langle x,u\rangle - \min_{\substack{v\in\prod\mathbb{R}^{G_r}\\ P^*v=x}} \sum_{r=1}^B \|v_r\|_p\right] = \sup_{x\in\mathbb{R}^d}\left[\sup_{\substack{v\in\prod\mathbb{R}^{G_r}\\ P^*v=x}} \langle x,u\rangle - \sum_{r=1}^B \|v_r\|_p\right] \\
&= \sup_{v\in\prod\mathbb{R}^{G_r}}\left[\langle\sum_{r=1}^B P_r^* v_r, u\rangle - \sum_{r=1}^B \|v_r\|_p\right] = \sum_{r=1}^B \sup_{v_r\in\mathbb{R}^{G_r}}\left[\langle P_r^* v_r, u\rangle - \|v_r\|_p\right] \\
&\sum_{r=1}^B \sup_{v_r\in\mathbb{R}^{G_r}}\left[\langle v_r, P_r u\rangle - \|v_r\|_p\right] = \sum_{r=1}^B \iota_q(P_r u),
\end{aligned}$$

where ι_q is the Fenchel conjugate of $\|\cdot\|_p$, i.e. the indicator function of the ℓ_q unitary ball in $\mathbb{R}^{G_r}$. We can rewrite the sum of indicator functions as $\sum_{r=1}^{B} \iota_q(P_r u) = \iota_{K_p^{\mathcal{G}}}(u)$. It is well-known that

$$\text{prox}_{\lambda \iota_{K_p^{\mathcal{G}}}}(x) = \pi_{K_p^{\mathcal{G}}}(x).$$

Using the Moreau decomposition (5) and basic properties of the projection we obtain

$$\text{prox}_{\lambda\Omega}(x) = x - \lambda \pi_{K_p^{\mathcal{G}}}(x/\lambda) = x - \pi_{\lambda K_p^{\mathcal{G}}}(x). \tag{6}$$

□

The following lemma shows that, when evaluating the projection $\pi_{K_p^{\mathcal{G}}}(x)$, we can restrict ourselves to a subset of *active* groups, denoted by $\hat{\mathcal{G}} = \mathcal{G}(\hat{x})$ and defined in Lemma 2. This equivalence is crucial to speed up Algorithm 1, in fact the number of active groups at iteration m will converge to the number of selected groups, which is typically small if one is interested in sparse solutions.

Lemma 2. *Given $x \in \mathbb{R}^d$, it holds*

$$\pi_{\lambda K_p^{\mathcal{G}}}(x) = \pi_{\lambda K_p^{\hat{\mathcal{G}}}}(x)\,, \tag{7}$$

where $\hat{\mathcal{G}} := \left\{ G \in \mathcal{G},\ \|x\|_{G,q} > \lambda \right\}$.

Proof. Given a group of indices G and a number $p > 1$, we denote by $C_{G,p}$ the convex set

$$C_{G,p} = \{x \in \mathbb{R}^d \,:\, \|x\|_{G,q} \leq 1\}.$$

To prove the result we first show that for any subset $\mathcal{S} \subseteq \mathcal{G}$ the projection onto the intersection $\lambda K_p^{\mathcal{S}} = \cap_{G \in \mathcal{S}} \lambda C_{G,p}$ is non-expansive coordinate-wise with respect to zero. More precisely, for all $x \in \mathbb{R}^d$, it holds that $|\pi_{\lambda K_p^{\mathcal{S}}}(x)_i| \leq |x_i|$ for all $i = 1, \ldots, d$ and for all $\lambda > 0$. By contradiction, assume that there exists an index $\hat{j}$ such that $|\pi_{\lambda K_p^{\mathcal{S}}}(x)_{\hat{j}}| > |x_{\hat{j}}|$. Consider the vector $\tilde{x}$ defined by setting

$$\tilde{x}_j = \begin{cases} \pi_{\lambda K_p^{\mathcal{S}}}(x)_j & \text{if } j \neq \hat{j} \\ x_{\hat{j}} & \text{otherwise.} \end{cases}$$

First note that $\tilde{x} \in \lambda K_p^{\mathcal{S}}$, since $\|\tilde{x}\|_{G,q} \leq \left\| \pi_{\lambda K_p^{\mathcal{S}}}(x) \right\|_{G,q} \leq \lambda$ for all $G \in \mathcal{S}$. On the other hand

$$\|x - \tilde{x}\|^2 = \sum_{\substack{j=1 \\ j \neq \hat{j}}}^{d} (x_j - \tilde{x}_j)^2 < \left\| x - \pi_{\lambda K_p^{\mathcal{S}}}(x) \right\|^2,$$

which is a contradiction. To conclude, suppose that $x \in \lambda K_p^{\mathcal{S}}$, with $\mathcal{S} \subseteq \mathcal{G}$. If we prove that

$$\pi_{\lambda K_p^{\mathcal{G}}}(x) = \pi_{\lambda K_p^{\mathcal{G}\setminus\mathcal{S}}}(x),$$

we are done. For the sake of brevity denote $v = \pi_{\lambda K_p^{\mathcal{G}\setminus\mathcal{S}}}(x)$. Thanks to the non-expansive property it follows $|v_j| \leq |x_j|$ for all $j = 1, \ldots, d$ and therefore $v \in \lambda K_p^{\mathcal{S}}$. Since $v \in \lambda K_p^{\mathcal{G}\setminus\mathcal{S}}$ by hypotheses, we get that $v \in \lambda K_p^{\mathcal{G}}$. Furthermore by definition of projection

$$\|v - x\| \leq \|w - x\|\,, \qquad \text{for every } w \in \lambda K_p^{\mathcal{G}\setminus\mathcal{S}}$$

and a fortiori $\|v - x\| \leq \|w - x\|$ for every $w \in \lambda K_p^{\mathcal{G}}$. □

3.2.1 The projection on $K_p^{\mathcal{G}}$ for general p

The convex set $K_p^{\mathcal{G}}$ is an intersection of convex sets, precisely

$$K_p^{\mathcal{G}} = \bigcap_{G \in \mathcal{G}} C_{G,p}$$

where $C_{G,p} = \{v \in \mathbb{R}^d, \|v\|_{G,q} \leq 1\}$.

For general p a possible minimization scheme for computing the projection in (7) can be obtained by applying the Cyclic Projections algorithm [8] or one of its modified versions (see [4] and references therein). In the particular case of $p = 2$, we describe the Lagrangian dual problem corresponding to the projection onto $K_2^{\mathcal{G}}$, and we propose an alternative optimization scheme, the projected Newton method [7], which better exploits the geometry of the set $K_2^{\mathcal{G}}$, and in practice proves to be faster than the Cyclic Projections algorithm. Note that, in order to satisfy the hypothesis of Theorem 14, the tolerance for stopping the iteration must decrease with the outer iteration m.

A simple way to compute the projection onto the intersection of convex sets is given by the *Cyclic Projections* algorithm [8], which amounts to cyclically projecting onto each set. Here we recall in Algorithm 2 a modification of the Cyclic Projections algorithm proposed by [4], for which strong convergence is guaranteed (see Theorem 4.1 in [4]).

Algorithm 2 Cyclic Projections

Given $x \in \mathbb{R}^d, \{C_{G_1,p}, \ldots, C_{G_B,p}\}$
Let $l = 0$, $w^0 = x$ and find $C_{\hat{G}_1,p}, \ldots, C_{\hat{G}_{\hat{B}},p}$
while convergence not reached **do**
$l = l + 1$
Let π_l the projection onto $\tau C_{\hat{G}_{l \bmod \hat{B}},p}$

$$w^l = \frac{1}{l+1} x + \frac{l}{l+1} \pi_l(w^{l-1})$$

end while

In the following we describe how to compute each projection $\pi_{C_{p,r}}$ for specific values of p.

$\mathbf{p = 2}$. In this case $q = 2$, and the projection is trivial

$$[\pi_{\tau C_{G,2}}(w)]_j = \begin{cases} \tau \frac{w_j}{\|w\|_{G,2}} & \text{if } j \in G \text{ and } \|w\|_{G,2} > \tau \\ w_j & \text{otherwise} \end{cases}$$

$\mathbf{p = \infty}$. In this case $q = 1$, and $C_{G,\infty}$ is an ℓ_1 ball when restricting to the coordinates in G. From Lemma 4.2 in [19], we have that if $\|w\|_1 > \tau$, then the projection of w onto the ℓ_1 ball of radius τ, τB_1, is given by the *soft-thresholding operation*

$$[\pi_{\tau B_1}(w)]_j = (|w_j| - \mu)_+ \text{sign}(w_j)$$

where μ (depending on w and τ) is chosen such that $\sum_j (|w_j| - \mu)_+ = \tau$.

We recall a simple procedure provided in [19] for determining μ. In a first step, sort the absolute values of the components of w, resulting in the rearranged sequence, $w_j^* \geq w_{j+1}^* \geq 0$ for all j. Next, perform a search to find k such that

$$\sum_{j=1}^{k-1} (w_j^* - w_k^*) \leq \tau \leq \sum_{j=1}^{k} (w_j^* - w_{k+1}^*).$$

Then set $\mu = w_k^* + k^{-1} \left(\sum_{j=1}^{k-1} (w_j^* - w_k^*) - \tau \right)$

$\mathbf{p \neq 2, +\infty}$. In these cases no known closed form for the projection on the set $C_{G_r,p}$ exist, but it can be efficiently computed using Newton's method, as done in [22].

3.2.2 The projection on $K_p^{\mathcal{G}}$ for $p = 2$

When $p = 2$, the projection onto $K_2^{\mathcal{G}}$ amounts to solving the constrained minimization problem

$$\begin{array}{ll} \text{Minimize} & \|v - x\|^2 \\ \text{subject to} & v \in \mathbb{R}^d,\ \|v\|_{G,2} \leq \tau,\ \text{ for } G \in \hat{\mathcal{G}}, \end{array} \tag{8}$$

which Lagrangian dual problem can be written in a closed form. Working on the dual is advantageous, since the number of groups is typically much smaller than d, and furthermore Lemma 2 guarantees that one can restrict to the subset of groups

$$\hat{\mathcal{G}} := \{G \in \mathcal{G} :\ \|x\|_{G,2} > \tau\} =: \{\hat{G}_1, \ldots, \hat{G}_{\hat{B}}\} \tag{9}$$

which in general is a proper subset of $\mathcal{G}$.

In the following theorem we show how to compute the solution to problem (8), by solving the associated dual problem.

Theorem 1. *Given $x \in \mathbb{R}^d$, $\mathcal{G} = \{G_r\}_{r=1}^B$ with $G_r \subset \{1, \ldots, d\}$, $\hat{\mathcal{G}}$ as in* (9) *and $\tau > 0$, the projection of x onto the convex set $\tau K_2^{\mathcal{G}}$ with $K_2^{\mathcal{G}} = \{v \in \mathbb{R}^d : \|v\|_{G_r,2} \leq \tau \text{ for } r = 1, \ldots, B\}$ is given by*

$$\left[\pi_{\tau K_2^{\mathcal{G}}}(x)\right]_j = \frac{x_j}{1 + \sum_{r=1}^{\hat{B}} \lambda_r^* \mathbf{1}_{r,j}} \qquad \text{for } j = 1, \ldots, d \tag{10}$$

where λ^* is the solution of

$$\underset{\lambda \in \mathbb{R}_+^{\hat{B}}}{\operatorname{argmax}} f(\lambda), \qquad \text{with} \quad f(\lambda) = \sum_{j=1}^{d} \frac{-x_j^2}{1 + \sum_{r=1}^{\hat{B}} \mathbf{1}_{r,j} \lambda_r} - \sum_{r=1}^{\hat{B}} \lambda_r \tau^2, \tag{11}$$

and $\mathbf{1}_{r,j}$ equal to 1 if j belongs to group $\hat{G}_r$ and 0 otherwise.

Proof. The Lagrangian function for the minimization problem (8) is defined as

$$\begin{aligned} L(v, \lambda) &= \|v - x\|^2 + \sum_{r=1}^{\hat{B}} \lambda_r (\|v\|_{G_r}^2 - \tau^2) \\ &= \sum_{j=1}^{d} \left[(v_j - x_j)^2 + \sum_{r=1}^{\hat{B}} \lambda_r \mathbf{1}_{r,j} v_j^2 \right] - \sum_{r=1}^{\hat{B}} \lambda_r \tau^2 \\ &= \sum_{j=1}^{d} (1 + \sum_{r=1}^{\hat{B}} \mathbf{1}_{r,j} \lambda_r) \left(v_j - \frac{x_j}{1 + \sum_{r=1}^{\hat{B}} \mathbf{1}_{r,j} \lambda_r} \right)^2 - \sum_{j=1}^{d} \frac{x_j^2}{1 + \sum_{r=1}^{\hat{B}} \mathbf{1}_{r,j} \lambda_r} - \sum_{r=1}^{\hat{B}} \lambda_r \tau^2 + \|x\|^2 \end{aligned} \tag{12}$$

where $\lambda \in \mathbb{R}^{\hat{B}}$. The dual function is then

$$f(\lambda) = \inf_{v \in \mathbb{R}^d} L(v, \lambda) = L\left(\frac{x_j}{1 + \sum_{r=1}^{\hat{B}} \mathbf{1}_{r,j} \lambda_r}, \lambda \right) = -\sum_{j=1}^{d} \frac{x_j^2}{1 + \sum_{r=1}^{\hat{B}} \mathbf{1}_{r,j} \lambda_r} - \sum_{r=1}^{\hat{B}} \lambda_r \tau^2 + \|x\|^2 .$$

Since strong duality holds, the minimum of (4) is equal to maximum of the dual problem which is therefore

$$\begin{array}{ll} \text{Maximize} & f(\lambda) \\ \text{subject to} & \lambda_r \geq 0 \text{ for } r = 1, \ldots, \hat{B}. \end{array} \tag{13}$$

Once the solution λ^* to the dual problem (13) is obtained, the solution to the primal problem (8), v^*, is given by

$$v_j^* = \frac{x_j}{1 + \sum_{r=1}^{\hat{B}} \lambda_r^* \mathbf{1}_{r,j}} \qquad \text{for } j = 1, \dots, d.$$

□

The dual problem can be efficiently solved, for instance, via Bertsekas' projected Newton method described in [7], and here reported as Algorithm 5 in the Appendix, where the first and second partial derivatives of $f(\lambda)$ are given by

$$\partial_r f(\lambda) = \sum_{j=1}^{d} \frac{x_j^2 \mathbf{1}_{r,j}}{(1 + \sum_{s=1}^{\hat{B}} \mathbf{1}_{s,j} \lambda_s)^2} - \tau^2,$$

and

$$\begin{aligned} \partial_r \partial_s f(\lambda) &= -\sum_{j=1}^{d} \frac{2 x_j^2 \mathbf{1}_{r,j} \mathbf{1}_{s,j}}{(1 + \sum_{s=1}^{\hat{B}} \mathbf{1}_{s,j} \lambda_s)^3} \\ &= \begin{cases} 0 & \text{if } \hat{G}_r \cap \hat{G}_s = \emptyset \\ -2 \sum_{j \in \hat{G}_r \cap \hat{G}_s} x_j^2 (1 + \sum_{s=1}^{\hat{B}} \mathbf{1}_{s,j} \lambda_s)^{-3} & \text{otherwise.} \end{cases} \end{aligned}$$

Bertsekas' iterative scheme combines the basic simplicity of the steepest descent iteration [43] with the quadratic convergence of the projected Newton's method [9]. It does not involve the solution of a quadratic program thereby avoiding the associated computational overhead. Its convergence properties have been studied in [7] and are briefly mentioned in next section.

3.3 Convergence analysis of GSO-p Algorithm

In this subsection we clarify the accuracy in the computation of the projection which is required to prove convergence of the Algorithm 1. As mentioned above, we rely on recent theorems providing a convergence rate for proximal gradient methods with approximations.

Definition 1. *We say that w is an approximation of $\pi_{\tau/\sigma K_p^{\mathcal{G}}}(x)$ with tolerance ϵ if $\|w - \pi_{\tau/\sigma K_p^{\mathcal{G}}}(x)\| \leq \epsilon$.*

Theorem 2. *Given $x^0 \in \mathbb{R}^d$, and $\sigma = ||\Psi^T \Psi||/n$. Assume that $\pi_{\tau/\sigma K_p^{\mathcal{G}}}(x^m)$ in Algorithm 1 is approximately computed at step m with tolerance $\epsilon_m = \epsilon_0/m^\alpha$.*

- *If $\alpha > 2$, there exists a constant $C_I := C_I(p, \mathcal{G}, x_0, \sigma, \tau, \alpha)$ such that the iterative update* (ISTA) *satisfies*

$$\mathcal{E}_\tau^p \left(\frac{1}{m} \sum_{i=1}^{m} x^i \right) - \mathcal{E}_\tau^p(x^*) \leq \frac{C_I}{m}. \tag{14}$$

- *If $\alpha > 4$, there exists a constant $C_F := C_F(p, \mathcal{G}, x_0, \sigma, \tau, \alpha)$ such that the iterative update* (FISTA) *satisfies*

$$\mathcal{E}_\tau^p (x^m) - \mathcal{E}_\tau^p(x^*) \leq \frac{C_F}{m^2}. \tag{15}$$

Proof. It is enough to show that there exists a constant $C > 0$ (independent of w^l and x^m) such that

$$\left\| w^l - \pi_{\tau K_p^{\mathcal{G}}}(x^m) \right\| \leq \frac{\epsilon_m}{C} \implies \Phi_{\frac{\tau}{\sigma}}(w^l) \leq \min \Phi_{\frac{\tau}{\sigma}} + \epsilon_m \tag{16}$$

where $\Phi_{\frac{\tau}{\sigma}}$ is defined as in (3). Then the statement directly follows from Proposition 1 and Proposition 2 in [45]. In order to prove equation (16) first note that thanks to the assumption $\cup_{r=1}^{B} G_r = \{1, \dots, d\}$ made at the beginning,

it easily follows from the definition that $\Omega_p^{\mathcal{G}}$ is a norm on $\mathbb{R}^d$, and therefore it is equivalent to the euclidean one. Thus, there exists a constant A (depending only on p and $\mathcal{G}$) such that

$$\Omega_p^{\mathcal{G}}(x) - \Omega_p^{\mathcal{G}}(x') \leq A \left\| x - x' \right\|, \qquad \forall\, x, x' \in \mathbb{R}^d.$$

Next, let w and x be such that

$$\left\| w - \pi_{\tau/\sigma K_p^{\mathcal{G}}}(x) \right\| \leq \gamma, \tag{17}$$

for some $\gamma > 0$ (and suppose w.l.o.g. that $\gamma < 1$). By Lemma 1 and by definition of $\operatorname{prox}_{\frac{\tau}{\sigma}\Omega_p^{\mathcal{G}}}(x)$ and $\Phi_{\frac{\tau}{\sigma}}$ (see equation (3)) we have

$$\Phi_{\frac{\tau}{\sigma}}(x - \pi_{\tau/\sigma K_p^{\mathcal{G}}}(x)) = \min \Phi_{\frac{\tau}{\sigma}}.$$

Thus, by equation (17), and using the fact that $\Omega_p^{\mathcal{G}}$ is a norm

$$\begin{aligned}
\Phi_{\frac{\tau}{\sigma}}(x - w) &= \frac{\sigma}{2\tau} \left\| w \right\|^2 + \Omega_p^{\mathcal{G}}(x - w) \\
&\leq \frac{\sigma}{2\tau} \left\| w - \pi_{\tau/\sigma K_p^{\mathcal{G}}}(x) \right\|^2 + \frac{\sigma}{2\tau} \left\| \pi_{\tau/\sigma K_p^{\mathcal{G}}}(x) \right\|^2 + \frac{\sigma}{\tau} \langle w - \pi_{\tau/\sigma K_p^{\mathcal{G}}}(x), \pi_{\tau/\sigma K_p^{\mathcal{G}}}(x) \rangle \\
&\quad + \Omega_p^{\mathcal{G}}(x - \pi_{\tau/\sigma K_p^{\mathcal{G}}}(x)) + \Omega_p^{\mathcal{G}}(\pi_{\tau/\sigma K_p^{\mathcal{G}}}(x) - w) \\
&\leq \min \Phi_{\frac{\tau}{\sigma}} + \frac{\sigma}{2\tau}\gamma^2 + \frac{\sigma}{\tau}\gamma\tilde{A} + A\gamma \\
&= \min \Phi_{\frac{\tau}{\sigma}} + \left(\frac{\sigma}{2\tau}\gamma + \frac{\sigma}{\tau}\tilde{A} + A \right) \gamma \\
&\leq \min \Phi_{\frac{\tau}{\sigma}} + C\gamma
\end{aligned}$$

where $\tilde{A}$ is such that $\sup_{v \in K_p^{\mathcal{G}}} \left\| v \right\| \leq \tilde{A}$ and $C = C(p, \mathcal{G}, \sigma, \tau)$. Therefore, equation (16) holds with C as defined above. □

As it happens for the exact accelerations of the basic forward-backward splitting algorithm such as [33, 6, 5], convergence of the sequence x^m is no longer guaranteed unless strong convexity is assumed.

By Theorem 3.1 in [4], Algorithm 2 is strongly convergent, and therefore, given arbitrary $\epsilon > 0$ and $x \in \mathbb{R}^d$, there exists an index $l_m := l_m(\epsilon)$ such that w^{l_m} produced through Algorithm 2 enjoys the property

$$\left\| w^l - \pi_{\tau K_p^{\mathcal{G}}}(x^m) \right\| \leq \epsilon,$$

for every $l \geq l_m$.

Algorithm 1 combined with Algorithm 2 thus converges to the minimum of (GSO-p) problem with rate $1/m^2$, if the projection is approximately computed with tolerance ϵ_0/m^α with $\alpha > 4$. Similarly, one can use ISTA instead of FISTA as updating rule in Algorithm 1, obtaining the convergence rate $1/m$, and setting $\alpha > 2$. It is clear that the choice of α defines the stopping rule for the internal algorithm (see Subsection 4.1).

Every other algorithm producing admissible approximations can be used in place of Algorithm 2 in the computation of the projection. In the case $p = 2$, we tested Bertsekas' projected Newton method, reported in the Appendix as Algorithm 5. Its convergence is not always guaranteed, since there are particular choices of x and $\mathcal{G}$ for which the partial Hessian of the dual function is not strictly positive defined, as would be required to ensure strong convergence (see Proposition 3 and Proposition 4 in [7]). However, ideas which are useful for circumventing the same problem for unconstrained Newton's method, such as preconditioning, could be easily adapted to this case, and convergence has always been observed in our experiments (for more details see the discussion in [7] and also the comments at the end of the next subsection).

3.4 Computing the regularization path

In Algorithm 3 we report the complete scheme for computing the regularization path for the Group-wise Selection with Overlap problem (GSO-p), i.e. the set of solutions corresponding to different values of the regularization parameter $\tau_1 > \ldots > \tau_T$. Note that we employ the *continuation* strategy proposed in [20]. When computing the

Algorithm 3 Regularization path for GSO-p

Given: $\tau_1 > \tau_2 > \cdots > \tau_T, \mathcal{G}, \epsilon_0 > 0, \nu > 0$
Let: $\sigma = ||\Psi^T \Psi||/n,\ x^{(\tau_0)} = 0$
for $t = 1, \ldots, T$ **do**
  Initialize: $x = x^{(\tau_{t-1})}$
  while `convergence not reached` **do**
    • update x according to Algorithm 1, with the projection computed via Cyclic Projections or by solving the dual problem
  end while
  $x^{(\tau_t)} = x$
end for
return $x^{(\tau_1)}, \ldots, x^{(\tau_T)}$

proximity operator with Bertsekas' projected Newton method, a similar warm starting is applied to the inner iteration, since the m-th projection is initialized with the solution of the $(m-1)$-th projection. Despite the local nature of Bertsekas' scheme, such an initialization empirically proved to guarantee convergence.

3.5 The replicates formulation

As discussed in Section 2, the most common method to solve (GSO-p) problem is to minimize the standard group ℓ_1/ℓ_p regularization (without overlap) in the expanded space of latent variables in (2) built by replicating variables belonging to more than one group, thus working in a $\tilde{d}$-dimensional space with $\tilde{d} = \sum_{r=1}^{B} |G_r|$. Setting $\tilde{\Psi} = \Psi P^*$ and $\mathrm{R}_p^{\mathcal{G}}(v) = \sum_{r=1}^{B} \|v_r\|_p$, problem (2) can be written as

$$\min_{v \in \prod_{r=1}^{B} \mathbb{R}^{G_r}} \frac{1}{n} \left\| \tilde{\Psi} v - y \right\|^2 + 2\tau \mathrm{R}_p^{\mathcal{G}}(v).$$

The main advantage of such a formulation relies on the possibility of using any state-of-the-art optimization procedure for ℓ_1/ℓ_p regularization without overlap. In terms of proximal methods, a possible solution is given by Algorithm 3, where the proximity operator can be now computed group-wise as

$$\left((\mathrm{prox}_{\lambda \mathrm{R}_p^{\mathcal{G}}}(v))_j \right)_{j \in G_r} = \left(I - \pi_{\lambda S_{G_r,p}} \right) \left((v_j)_{j \in G_r} \right)$$

for all $r = 1, \ldots, B$, where $S_{G_r,p}$ now denotes the ℓ_q unitary ball in $\mathbb{R}^{G_r}$. Furthermore for $p = 2$ and $p = +\infty$ each projection can be computed exactly as described in Subsection 3.2.1 , and the proximity operator of $\mathrm{R}_p^{\mathcal{G}}$ is thus exact. The optimization algorithm for solving (GSO-p) via FISTA in the replicated space is reported in Algorithm 4.

The replicate formulation involves a much simpler proximity operator, but each iteration has higher computational cost, since now depends on $\tilde{d}$ rather than on d, and thus increases with the amount of overlap among variables subsets (see Section 4 for numerical comparisons between the projection and replication approaches).

4 Numerical experiments

In this section we present numerical experiments aimed at studying the computational performance of the proposed family of optimization algorithms, and at comparing them with the state-of-the-art algorithms applied to the replicate formulation.

Algorithm 4 FISTA for Group-wise Selection without overlap

Given: $v^0 \in \prod_{r=1}^{B} \mathbb{R}^{G_r}, \tau > 0, \sigma = ||\tilde{\Psi}^T \tilde{\Psi}||/n$
Initialize: $m = 0, w^1 = v^0, t^1 = 1$
while `convergence not reached` **do**
  for $r = 1, \dots, B$ **do**

$$v_r = \left(I - \pi_{\frac{\tau}{\sigma} S_{G_r,p}}\right) \left(\left(w^m - \frac{1}{n\sigma} \tilde{\Psi}^T (\tilde{\Psi} w^m - y) \right)_{j \in G_r} \right)$$

  end for

$$s_{m+1} = \frac{1}{2} \left(1 + \sqrt{1 + 4 s_m^2} \right)$$

$$w^{m+1} = \left(1 + \frac{s_m - 1}{s_{m+1}} \right) v^m + \left(\frac{1 - s_m}{s_{m+1}} \right) v_{m-1}$$

end while
return v^m

4.1 Cyclic Projections vs dual formulation

We build B groups,$\{G_r\}_{r=1}^{B}$, of size b, with $G_r \subseteq \{1, \dots, d\}$, by randomly drawing sets of b indexes from $\{1, \dots, d\}$, and consider the cases $b = 10$, and $b = 100$. We vary the number of groups B, so that the dimension of the expanded space is α times the input dimension, $\tilde{d} = \alpha d$, with $\alpha = 1.2, 2$ and 5. Clearly this amounts to taking $B = \alpha \cdot d/b$. We then generate a vector $x \in \mathbb{R}^d$ by randomly drawing each of its entry from $\mathcal{N}(0, 1)$. We then pick a value of τ such that, when computing $\text{prox}_{\tau \Omega_p^{\mathcal{G}}}(x)$, all groups are active. Precisely we take $\tau = .8 \cdot \min_{r=1,\dots,B} \|x\|_{G_r,2}$. We first compute the *exact* solution $x^\dagger = \text{prox}_{\Omega_2^{\mathcal{G}}}(x)$ [1]. Then we compute the approximated solutions with the Cyclic Projections Algorithm 2 and by solving the dual via the projected Newton method. We will refer to the former as CP2 and to the latter as *dual*. We stop the iteration when the distance from the exact solution is less than ϵ the norm of $x^\dagger$. We consider different values for the tolerance ϵ, precisely we take $\epsilon = 10^{-2}, 10^{-3}, 10^{-4}$.

Mean and standard deviation of the computing time over 20 repetitions are plotted in Figure 1 and 2 for each value of α and ϵ. The dual formulation is faster than the Cyclic Projections algorithm in most situations. It is convenient to use Cyclic Projections when the number of active groups is high and the required tolerance very low. When computing the projection for Algorithm 1, it is thus reasonable to use Cyclic Projections in the very first outer iterations, when the tolerance – which depends on the outer iterations – is low, and the solution could be not sparse, because still far from convergence. After few iterations, it is more convenient to resort to the dual formulation. Even though, not optimal, in the following experiments, when denoting GSO-2 via projection we will consider always the projection computed with the dual formulation.

4.2 Projection vs replication

In this Subsection we compare the running time performance of the proposed set of algorithms where the proximity operator is computed approximately, to state-of-the-art algorithms used to solve the equivalent formulation in the replicated space. For such a comparison we restrict to $p = 2$, since many benchmark algorithms are available in the case of groups that do not overlap. In order to ensure a fair comparison, we first run some preliminary experiments to identify the fastest codes for group ℓ_1 regularization with no overlap.

[1] it is the solution computed via the projected Newton method for the dual problem with very tight tolerance

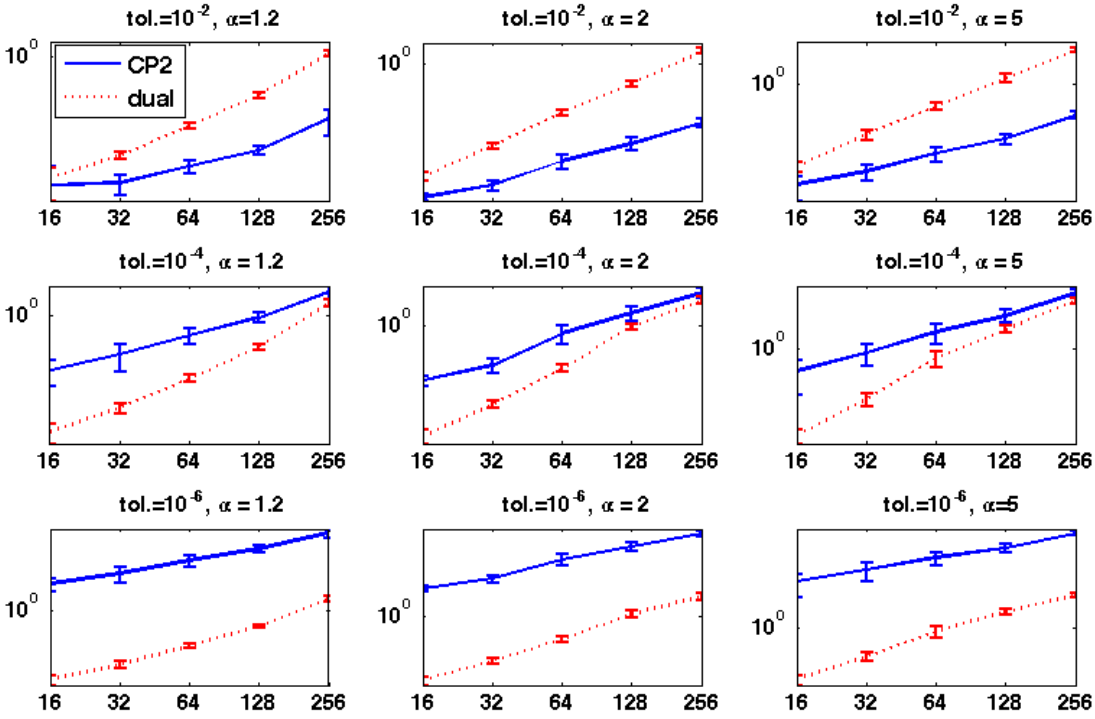

Figure 1: Computing time (in seconds) necessary for evaluating the prox vs number of variables (d), for different values of the overlap degree α and the tolerance, for fixed group size $b = 10$.

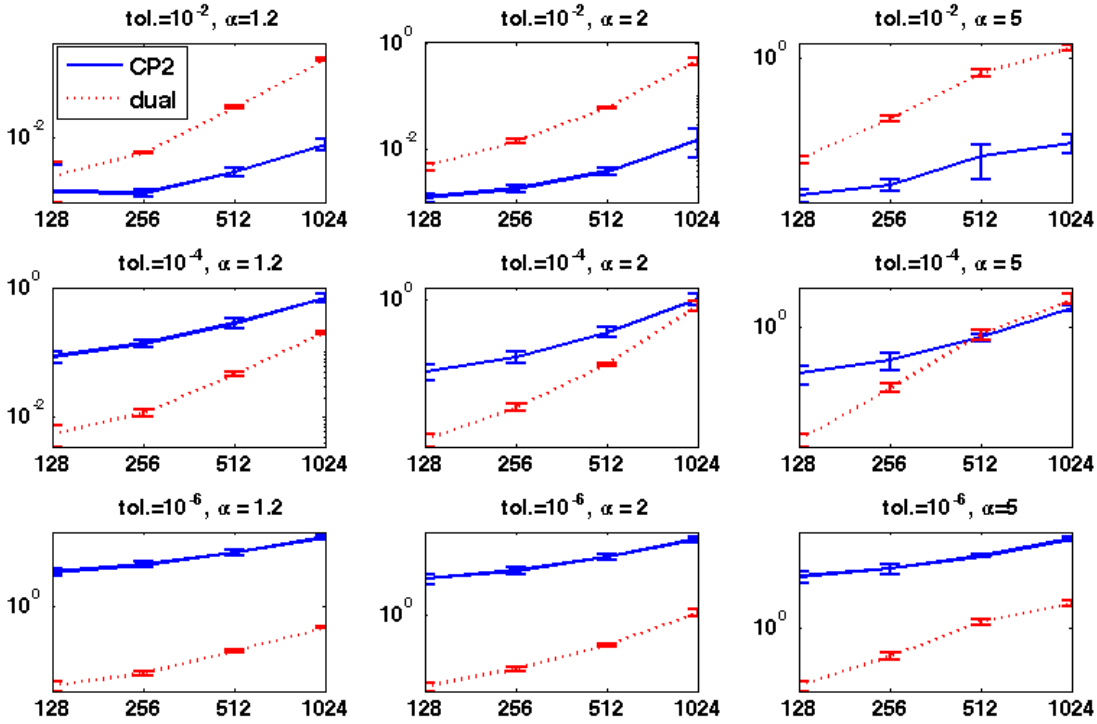

Figure 2: Computing time (in seconds) necessary for evaluating the prox vs number of variables (d), for different values of the overlap degree α and the tolerance, for fixed group size $b = 100$

4.2.1 Comparison without overlap

Recently there has been a very active research on this topic, see e.g. [39, 40, 12]. For the comparison, we considered three algorithms which are representative of the optimization techniques used to solve group lasso: interior-point methods, (group) coordinate descent and its variations, and proximal methods. As an instance of the first set of techniques we employed the publicly available Matlab code at `http://www.di.ens.fr/~fbach/grouplasso/index.htm` described in [1]. For coordinate descent methods, we employed the R-package `grlplasso`, which implements block coordinate gradient descent minimization for a set of possible loss functions. In the following we will refer to these two algorithms as "IP" and "BCGD". Finally, as an instance of proximal methods, we use our Matlab implementation of FISTA for Group-wise Selection, namely Algorithm 4 with FISTA instead of ISTA as updating rule. We will refer to it as "PROX".

We first observe that the solutions of the three algorithms coincide up to an error which depends on each algorithm tolerance. We thus need to tune the each tolerance in order to guarantee that all iterative algorithms are stopped when the level of approximation to the true solution is the same. Toward this end, we run Algorithm PROX with machine precision, $\nu = 10^{-16}$, in order to have a good approximation of the asymptotic solution. We observe that for many values of n and d, and over a large range of values of τ, the approximation of PROX when $\nu = 10^{-6}$ is of the same order of the approximation of IP with `optparam.tol` $= 10^{-9}$, and of BCGD with `tol` $= 10^{-12}$. Note also that with these tolerances the three solutions coincide also in terms of selection, i.e. their supports are identical for each value of τ. Therefore the following results correspond to `optparam.tol` $= 10^{-9}$ for IP, `tol` $= 10^{-12}$ for BCGD, and $\nu = 10^{-6}$ for PROX. For the other parameters of IP we used the values used in the demos supplied with the code.
Concerning the data generation protocol, the input variables $x = (x_1, \ldots, x_d)$ are uniformly drawn from $[-1, 1]^d$. The labels y are computed using a noise-corrupted linear regression function, i.e. $y = x \cdot x + w$, where x depends on the first 30 variables, $x_j = c$ if $j = 1, \ldots, 30$, and 0 otherwise, w is an additive noise, $w \sim N(0, 1)$, and c is a rescaling factor that sets the signal to noise ratio to 5:1. In this case the dictionary coincides with the variables, $\Psi_j(x) = x_j$ for $j = 1, \ldots, d$. We then evaluate the entire regularization path for the three algorithms with B sequential groups of 10 variables, (G_1=$[1, \ldots, 10]$, G_2=$[11, \ldots, 20]$, and so on), for different values of n and B. In order to make sure that we are working on the correct range of values for the parameter τ, we first evaluate the set of solutions of PROX corresponding to a large range of 500 values for τ, with $\nu = 10^{-4}$. We then determine the smallest value of τ which corresponds to selecting less than n variables, τ_{min}, and the smallest one returning the null solution, τ_{max}. Finally we build the geometric series of 50 values between τ_{min} and τ_{max}, and use it to evaluate the regularization path on the three algorithms. In order to obtain robust estimates of the running times, we repeat 20 times for each pair n, B.

In Table 1 we report the computational times required to evaluate the entire regularization path for the three algorithms. Algorithms BCGD and PROX are always faster than IP which, due to memory reasons, cannot be applied to problems where the number of variables are more than 5000, since it requires to store the $d \times d$ matrix $\Psi \times \Psi$. It must be said that the code for GP-IL was made available mainly in order to allow reproducibility of the results presented in [1], and is not optimized in terms of time and memory occupation. However it is well known that standard second-order methods are typically precluded on large data sets, since they need to solve large systems of linear equations to compute the Newton steps. PROX is the fastest for $B = 10, 100$ and has a similar behavior to BCGD. The candidates as benchmark algorithms for comparison with FISTA via projection are therefore BCGD and PROX. Since we are more familiar with the PROX algorithm, we therefore compare FISTA via projection with the PROX algorithm, i.e. FISTA via replication only.

4.2.2 Comparison with overlap

Here we compare two different implementations of the GSO-2 solution: FISTA via approximated projection computed by solving the dual problem with projected Newton method, and FISTA via replication. We will refer to the former as FISTA-proj, and to the latter as FISTA-repl.

The data generation protocol is equal to the one described in the previous experiments, but x depends on the

Table 1: Running time (mean and standard deviation) in seconds for computing the entire regularization path of IP, BCGD, and PROX for different values of B, and n.

$n = 100$	$B = 10$	$B = 100$
IP	5.6 ± 0.6	60 ± 90
BCGD	2.1 ± 0.6	2.8 ± 0.6
PROX	0.21 ± 0.04	2.9 ± 0.4

$n = 500$	$B = 10$	$B = 100$
IP	2.30 ± 0.27	370 ± 30
BCGD	2.15 ± 0.16	4.7 ± 0.5
PROX	0.1514 ± 0.0025	2.54 ± 0.16

$n = 1000$	$B = 10$	$B = 100$
IP	1.92 ± 0.25	328 ± 22
BCGD	2.06 ± 0.26	18 ± 3
PROX	0.182 ± 0.006	4.7 ± 0.5

first $12/5b$ variables (which correspond to the first three groups)

$$x = (\underbrace{c, \ldots, c}_{b \cdot 12/5 \text{ times}}, \underbrace{0, 0, \ldots, 0}_{d - b \cdot 12/5 \text{ times}}).$$

We then define B groups of size b, so that $\tilde{d} = B \cdot b > d$. The first three groups correspond to the subset of relevant variables, and are defined as $G_1 = [1, \ldots, b]$, $G_2 = [4/5b + 1, \ldots, 9/5b]$, and $G_3 = [1, \ldots, b/5, 8/5b + 1, \ldots, 12/5b]$, so that they have a 20% pair-wise overlap. The remaining $B - 3$ groups are built by randomly drawing sets of b indexes from $\{1, d\}$. In the following we will let $n = 10|G_1 \cup G_2 \cup G_3|$, i.e. n is ten times the number of relevant variables, and vary d, b. We also vary the number of groups B, so that the dimension of the space of latent variables is α times the input dimension, $\tilde{d} = \alpha d$, with $\alpha = 1.2, 2, 5$. Clearly this amounts to taking $B = \alpha \cdot d/b$. The parameter α can be thought of as the average number of groups a single variable belongs to. We identify the correct range of values for τ as in the previous experiments, using FISTA-proj with loose tolerance, and then evaluate the running time and the number of iterations necessary to compute the entire regularization path for FISTA-repl on the expanded space and FISTA-proj, both with $\nu = 10^{-6}$. Finally we repeat 20 times for each combination of the three parameters d, b, and α.

Table 2: Running time (mean $\pm$ standard deviation) in seconds for $b{=}10$ (top), and $b{=}100$ (below). For each d and α, the left and right side correspond to FISTA-proj, and FISTA-repl, respectively.

	$\alpha = 1.2$		$\alpha = 2$		$\alpha = 5$	
d=1000	0.15 ± 0.04	0.20 ± 0.09	1.6 ± 0.9	5.1 ± 2.0	12.4 ± 1.3	68 ± 8
d=5000	1.1 ± 0.4	1.0 ± 0.6	1.55 ± 0.29	2.4 ± 0.7	103 ± 12	790 ± 57
d=10000	2.1 ± 0.7	2.1 ± 1.4	3.0 ± 0.6	4.5 ± 1.4	460 ± 110	2900 ± 400

	$\alpha = 1.2$		$\alpha = 2$		$\alpha = 5$	
d=1000	11.7 ± 0.4	24.1 ± 2.5	11.6 ± 0.4	42 ± 4	13.5 ± 0.7	1467 ± 13
d=5000	31 ± 13	38 ± 15	90 ± 5	335 ± 21	85 ± 3	1110 ± 80
d=10000	16.6 ± 2.1	13 ± 3	90 ± 30	270 ± 120	296 ± 16	–

Table 3: Number of iterations (mean $\pm$ standard deviation) for $b = 10$ (top) and $b = 100$ (below). For each d and α, the left and right side correspond to FISTA-proj, and FISTA-repl, respectively.

	$\alpha = 1.2$		$\alpha = 2$		$\alpha = 5$	
d=1000	100 ± 30	80 ± 30	1200 ± 500	1900 ± 800	2150 ± 160	11000 ± 1300
d=5000	100 ± 40	70 ± 30	148 ± 25	139 ± 24	6600 ± 500	27000 ± 2000
d=10000	100 ± 30	70 ± 40	160 ± 30	137 ± 26	13300 ± 1900	49000 ± 6000

	$\alpha = 1.2$		$\alpha = 2$		$\alpha = 5$	
d=1000	913 ± 12	2160 ± 210	894 ± 11	2700 ± 300	895 ± 10	4200 ± 400
d=5000	600 ± 400	600 ± 300	1860 ± 110	4590 ± 290	1320 ± 30	6800 ± 500
d=10000	81 ± 11	63 ± 11	1000 ± 500	1800 ± 900	2100 ± 60	–

Running times and number of iterations are reported in Table 2 and 3, respectively. When the overlap, that is α, is low the computational times of FISTA-repl and FISTA-proj are comparable. As α increases, there is a clear advantage in using FISTA-proj instead of FISTA-repl. The same behavior occurs for the number of iterations.

4.3 $p = 2$ **vs** $p = \infty$

We generate the groups and the coefficient vector as in Subsection 4.1, with $b = 10$. Differently from the Subsection 4.1, here we compare the computational performance of the same algorithm applied to two different problems: Cyclic Projections for $p = 2$ and Cyclic Projections for $p = \infty$, that yield different solutions, since $\text{prox}_{\tau\Omega_2^{\mathcal{G}}}(x) \neq \text{prox}_{\tau\Omega_\infty^{\mathcal{G}}}(x)$. In order to guarantee a fair comparison we consider two different values of τ, τ_2 and τ_∞, such that, when computing $\text{prox}_{\tau_2\Omega_2^{\mathcal{G}}}(x)$ and $\text{prox}_{\tau_\infty\Omega_\infty^{\mathcal{G}}}(x)$, all groups are active. Precisely we take $\tau_2 = .8 \cdot \min_{r=1,\dots,B} \|x\|_{G_r,2}$. and $\tau_\infty = .8 \cdot \min_{r=1,\dots,B} \|x\|_{G_r,\infty}$. We compute the approximated solutions with the Cyclic Projections Algorithm 2 for $p = 2$ and $p = \infty$. We will refer to the former as CP2 and to the latter as CPinf. We stop the iteration when the relative decrease of the approximated solution is below ϵ. We consider different values for the tolerance ϵ, precisely we take $\epsilon = 10^{-2}, 10^{-3}, 10^{-4}$.

For each value of α and ϵ we estimate the number of iterations, and the computing time for the two algorithms, and average over 20 repetitions. Mean and standard deviation of number of iterations and the computing time are plotted in Figure 3 and 4. In all conditions CP2 is much faster than CPinf.

4.4 Real data Experiments: Microarray data

In the previous subsection we have shown that, thanks to the computational efficiency of the proposed projection algorithm, the GSO-p regularization scheme can be easily applied to large data sets with large group overlap. Here we show that on real data, indeed, dealing with the entire data set without resorting to preprocessing leads to improved prediction and selection performance. We consider the microarray experiment presented in [21] where the breast cancer dataset compiled by [49] (8141 genes for 295 tumors) is analyzed with the group lasso with overlap penalty and the 637 gene groups corresponding to the MSigDB pathways [46]. In [21] the accuracy of a logistic regression is estimated via 3-fold cross validation. On each split the 300 genes most correlated with the output are selected and the optimal τ is chosen via cross validation. 6, 5 and 78 pathways are selected with a 0.36 ± 0.03 cross validation error. We applied FISTA-proj to the entire data set with two loops of k-fold cross validation (k= 3 for testing). The obtained cross validation error is 0.33 ± 0.05 and 0.30 ± 0.06, with k= 3 and k= 10 for validation, respectively. In both cases the number of selected groups is $2, 3$, and 4, with 1 group in 3, and 3 pathways selected in 2 out of 3 splits. The computing time for running the entire framework for FISTA-proj (comprising data and pathways loading, recentering, selection via FISTA-proj, regression via RLS on the selected genes, and testing) is 850s (k=3) and 3387s (k=10). Note that, while the improved cross validation error might be due to the second optimization step (RLS), the improved stability is probably due to the absence of the

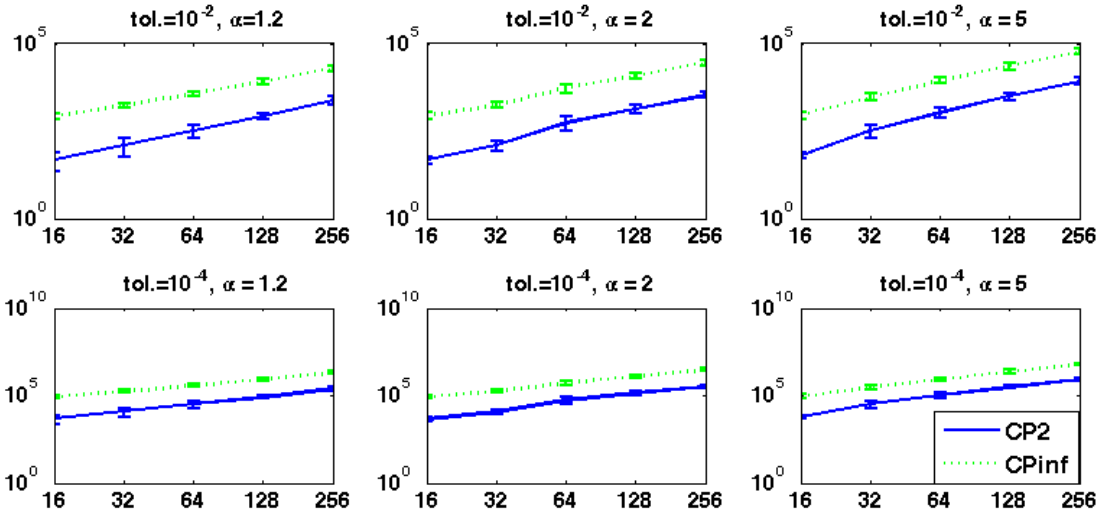

Figure 3: Number of iteration necessary for evaluating the prox vs number of variables (d), for different values of the overlap degree α, and the tolerance.

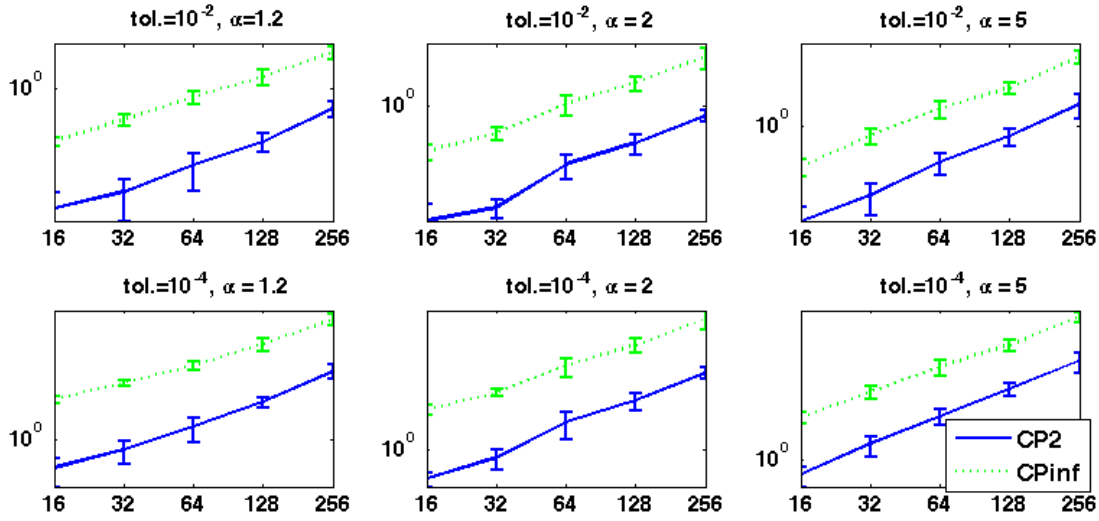

Figure 4: Computing time (in seconds) necessary for evaluating the prox vs number of variables (d), for different values of the overlap degree α, and the tolerance

preprocessing step, which can be highly unstable, thus compromising the overall stability of the solution.

5 Discussion

We have presented an efficient optimization procedure for computing the solution of a set of regularization schemes that perform group-wise selection with overlapping groups, whose convergence is guaranteed. Our procedure allows dealing with high dimensional problems with large group overlap. We have empirically shown that it has a significant computational advantage with respect to state-of-the-art algorithms for group-wise selection applied on the expanded space built by replicating variables belonging to more than one group. We also mention that computational performance may improve if our scheme is used as core for the optimization step of active set methods, such as [44]. Finally, the improved computational performance enables to use group-wise selection with overlap for pathway analysis of high-throughput biomedical data, since it can be applied to the entire data set and using all the information present in online databases, without pre-processing for dimensionality reduction.

Acknowledgments

Lorenzo Rosasco is assistant professor at DIBRIS, Università di Genova, Italy and currently on leave of absence. The authors wish to thank Saverio Salzo for carefully reading the paper.

References

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

[2] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4(1):1–106, 2012.

[3] F. R. Bach, G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *ICML*, volume 69 of *ACM International Conference Proceeding Series*, 2004.

[4] H. Bauschke. The approximation of fixed points of compositions of nonexpansive mappings in hilbert space. *Journal of Mathematical Analysis and its Applications*, 201(1):150–159, 1994.

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

[6] S. Becker, J. Bobin, and E. Candes. NESTA: A fast and accurate first-order method for sparse recovery. *SIAM J. on Imaging Sciences*, 4(1):1–39, 2009.

[7] D. Bertsekas. Projected newton methods for optimization problems with simple constraints. *SIAM Journal on Control and Optimization*, 20(2), 1982.

[8] J. Boyle and R. Dykstra. A method for finding projections onto the intersection of convex stes in hilbert spaces. In R. Dykstra, T. Robertson, and F. Wright, editors, *Advances in Order Restricted Statistical Inference*, volume 37 of *Lecture Notes in Statistics*, pages 28–48. Springer–Verlag, 1985.

[9] R. Brayton and J. Cullum. An algorithm for minimizing a differentiable function subject to. *J. Opt. Th. Appl.*, 29:521–558, 1979.

[10] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Math. Imaging Vision*, 40(1):120–145, 2011.

[11] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999.

[12] X. Chen, Q. Lin, S. Kim, J. Carbonell, and E.P. Xing. Smoothing proximal gradient method for general structured sparse regression. *Annals of Applied Statistics*, 2012.

[13] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Model. Simul.*, 4(4):1168–1200 (electronic), 2005.

[14] The Gene Ontology Consortium. Gene ontology: tool for the unification biology. *Nature Genetics*, 25:25 – 29, 2000.

[15] W. Deng, W. Yin, and Y. Zhang. Group sparse optimization by alternating direction method, 2011.

[16] J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2899–2934, December 2009.

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

[18] M. Fornasier, editor. *Theoretical Foundations and Numerical Methods for Sparse Recovery*, volume 9 of *Radon Series on Computational and Applied Mathematics*. De Gruyter, 2010.

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

[20] E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for l1-minimization: Methodology and convergence. *SIOPT*, 19(3):1107–1130, 2008.

[21] L. Jacob, G. Obozinski, and J.-P. Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th Intearnational Annual Conference on Machine Learning*, pages 433–440, 2009.

[22] L. Jacques, D. Hammond, and J. Fadili. Dequantizing compressed sensing: when oversampling and non-Gaussian constraints combine. *IEEE Trans. Inform. Theory*, 57(1):559–571, 2011.

[23] R. Jenatton, J.-Y . Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. Technical report, INRIA, 2009.

[24] R. Jenatton, G. Obozinski, and F. Bach. Structured principal component analysis. In *Proceedings of the 13 International Conference on Artificial Intelligence and Statistics*, May 2010.

[25] J. Liu and J. He. Fast overlapping group lasso, 2010.

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

[27] L. Meier, S. van de Geer, and P. Buhlmann. The group lasso for logistic regression. *J. R. Statist. Soc*, B(70):53–71, 2008.

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

[29] J.-J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *C. R. Acad. Sci. Paris*, 255:2897–2899, 1962.

[30] S. Mosci, L. Rosasco, M. Santoro, A. Verri, and S. Villa. Solving structured sparsity regularization with proximal methods. In J. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 6322 of *Lecture Notes in Computer Science*, pages 418–433. Springer, 2010.

[31] S. Mosci, S. Villa, A. Verri, and L. Rosasco. A primal-dual algorithm for group sparse regularization with overlapping groups. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2604–2612. 2010.

[32] Y. Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$. *Doklady AN SSSR*, 269(3):543–547, 1983.

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

[34] Y. Nesterov. Gradient methods for minimizing composite objective function. CORE Discussion Paper 2007/76, Catholic University of Louvain, September 2007.

[35] G. Obozinski, L. Jacob, and J.-P. Vert. Group Lasso with Overlaps: the Latent Group Lasso approach. Research report, October 2011.

[36] M. Y. Park and T. Hastie. L1-regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69:659–677, 2007.

[37] G. Peyré and J. Fadili. Group sparsity with overlapping partition functions. In *Proc. EUSIPCO 2011*, pages 303–307, 2011.

[38] E. Poliak. *Computational Methods in Optimization: A Unified Approach*. Academic Press, New York., 1971.

[39] Z. Qin and D. Goldfarb. Structured sparsity via alternating direction methods. *JMLR*, 13:1435–1468, 2012.

[40] Z. Qin, K. Scheinberg, and D. Goldfarb. Efficient block-coordinate descent algorithms for the group lasso, 2012.

[41] R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM J. Control Optimization*, 14(5):877–898, 1976.

[42] L. Rosasco, M. Mosci, S. Santoro, A. Verri, and S. Villa. Iterative projection methods for structured sparsity regularization. Technical Report MIT-CSAIL-TR-2009-050, MIT, 2009.

[43] J. Rosen. The gradient projection method for nonlinear programming, part i: linear constraints. *J. Soc. Ind. Appl. Math.*, 8:181–217, 1960.

[44] V. Roth and B. Fischer. The group-lasso for generalized linear models: uniqueness of solutions and efficient. In *Proceedings of 25th ICML*, 2008.

[45] M. Schmidt, N. Le Roux, and F. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *in Advances in Neural Information Processing Systems (NIPS)*, 2011.

[46] A. Subramanian and et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *PNAS*, 102(43), 2005.

[47] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.*, 58 No. 1:267–288, 1996.

[48] P. Tseng. Approximation accuracy, gradient methods, and error bound for structured convex optimization. *Math. Program.*, 125(2, Ser. B):263–295, 2010.

[49] L. van 't Veer et al. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871), 2002.

[50] S. Villa, S. Salzo, L. Baldassarre, and A. Verri. Accelerated and inexact forward-backward algorithms. *Optimization Online*, E-Print 2011 08 3132, 2011.

[51] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68(1):49–67, 2006.

[52] 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.

A Projected Newton Method

In this appendix we report as Algorithm 5 Bertsekas' projected Newton method described in [7], with the modifcations needed to perform the maximization of a concave function instead of the minimization of a convex one.

Algorithm 5 Projection onto $K_2^{\mathcal{G}}$

Given: $x \in \mathbb{R}^d, \lambda_{\text{init}} \in \mathbb{R}^{\hat{B}}, \eta \in (0,1), \delta \in (0,1/2), \epsilon > 0$
Initialize: $l = 0, \lambda^0 = \lambda^{\text{init}}$
while ($\partial_r f(\lambda^l) \neq 0$ `if` $\lambda_r > 0$, `or` $\partial_r f(\lambda^l) > 0$ `if` $\lambda_r = 0$, `for some` $r = 1, \dots, \hat{B}$) **do**
$l := l + 1$

$$\epsilon_l = \min\{\epsilon, ||\lambda^l - [\lambda^l + \nabla f(\lambda^l)]_+||\}$$
$$\mathcal{I}_+^l = \left\{r \,:\, 0 \leq \lambda_r^l \leq \epsilon_l,\ \partial_r f(\lambda^l) < 0\right\}$$
$$H_{r,s}^l = \begin{cases} 0 & \text{if } r \neq s, \text{and } r \in \mathcal{I}_+^l \text{or } s \in \mathcal{I}_+^l \\ \partial_r \partial_s f(\lambda^l) & \text{otherwise} \end{cases} \tag{18}$$
$$\lambda(\alpha) = [\lambda^l - \alpha (H^l)^{-1} \nabla f(\lambda^l)]_+$$

$m = 0$
while $f(\lambda(\eta^m)) - f(\lambda^l) < \delta \left\{ -\eta^m \sum_{r \notin \mathcal{I}_+^l} \sum_{s=1}^{\hat{B}} \partial_r f(\lambda^l) [(H^l)^{-1}]_{r,s} \partial_s f(\lambda^l) + \sum_{r \in \mathcal{I}_+^l} \partial_r f(\lambda^l) [\lambda_r(\eta^m) - \lambda_r^l] \right\}$ **do**
$m := m + 1$
end while

$$\lambda^{l+1} = \lambda(\eta^m)$$

end while
return λ^{l+1}

The step size rule, i.e. the choice of α, is a combination of the Armijo-like rule [43] and the Armijo rule usually employed in unconstrained minimization (see, e.g., [38]).