Sur la géométrie de problèmes d'optimisation et leur structure

Vincent Roulet

THÈSE DE DOCTORAT

de l'Université de recherche Paris Sciences Lettres
PSL Research University

Préparée à l'École normale supérieure

On the geometry of optimization problems and their structure

Sur la géométrie de problèmes d'optimisation et leur structure

École doctorale n°386

ÉCOLE DOCTORALE DE SCIENCES MATHÉMATIQUES DE PARIS CENTRE

Spécialité MATHÉMATIQUES APPLIQUÉES

COMPOSITION DU JURY :

M François Glineur
UCL Louvain-La-Neuve, Rapporteur

M Jérôme Malick
CNRS Grenoble, Rapporteur

M Alexandre d'Aspremont
CNRS Paris, Directeur de thèse

M Francis Bach
INRIA Paris, Président du Jury

M Jérôme Bolte
TSE Toulouse, Membre du Jury

M Zaid Harchaoui
UW Seattle, Membre du Jury

Soutenue par Vincent Roulet
le 21.12.2017

Dirigée par **Alexandre d'Aspremont**

ENS ÉCOLE NORMALE SUPÉRIEURE | PSL RESEARCH UNIVERSITY PARIS

Dédicacée à ma mère, mon père,
mes soeurs et mes grands-parents.

Abstract

In numerous fields such as machine learning, operational research or circuit design, a task is modeled by a set of parameters to be optimized in order to take the best possible decision. Formally, the problem amounts to minimize a function describing the desired objective with iterative algorithms. The development of these latter depends then on the characterization of the geometry of the function or the structure of the problem.

In a first part, this thesis studies how sharpness of a function around its minimizers can be exploited by restarting classical algorithms. Optimal schemes are presented for general convex problems. They require however a complete description of the function that is rarely available. Adaptive strategies are therefore developed and shown to achieve nearly optimal rates. A specific analysis is then carried out for sparse problems that seek for compressed representation of the variables of the problem. Their underlying conic geometry, that describes sharpness of the objective, is shown to control both the statistical performance of the problem and the efficiency of dedicated optimization methods by a single quantity.

A second part is dedicated to machine learning problems. These perform predictive analysis of data from large set of examples. A generic framework is presented to both solve the prediction problem and simplify it by grouping either features, samples or tasks. Systematic algorithmic approaches are developed by analyzing the geometry induced by partitions of the data. A theoretical analysis is then carried out for grouping features by analogy to sparse methods.

Keywords : Convex optimization, Error bound, Sparsity, Sharpness, Structured models with partitions of the data.

Résumé

Dans de nombreux domaines tels que l'apprentissage statistique, la recherche opérationnelle ou encore la conception de circuits, une tâche est modélisée par un jeu de paramètres que l'on cherche à optimiser pour prendre la meilleure décision possible. Mathématiquement, le problème revient à minimiser une fonction de l'objectif recherché par des algorithmes itératifs. Le développement de ces derniers dépend alors de la géométrie de la fonction ou de la structure du problème.

Dans une première partie, cette thèse étudie comment l'acuité d'une fonction autour de ses minima peut être exploitée par le redémarrage d'algorithmes classiques. Les schémas optimaux sont présentés pour des problèmes convexes généraux. Ils nécessitent cependant une description complète de la fonction, ce qui est rarement disponible. Des stratégies adaptatives sont donc développées et prouvées être quasi-optimales. Une analyse spécifique est ensuite conduite pour les problèmes parcimonieux qui cherchent des représentations compressées des variables du problème. Leur géométrie conique sous-jacente, qui décrit l'acuité de la fonction de l'objectif, se révèle contrôler à la fois la performance statistique du problème et l'efficacité des procédures d'optimisation par une seule quantité.

Une seconde partie est dédiée aux problèmes d'apprentissage statistique. Ceux-ci effectuent une analyse prédictive de données à l'aide d'un large nombre d'exemples. Une approche générique est présentée pour à la fois résoudre le problème de prédiction et le simplifier en groupant soit les variables, les exemples ou les tâches. Des méthodes algorithmiques systématiques sont développées en analysant la géométrie induite par une partition des données. Une analyse théorique est finalement conduite lorsque les variables sont groupées par analogie avec les méthodes parcimonieuses.

Mots-clés : Optimisation convexe, Borne d'erreur, Parcimonie, Acuité, Modèles structurés par partitions des données.

Remerciements

En avril 2014, lors de notre première rencontre, Alexandre, tu me proposais un "long-shot". Il dura en effet trois ans et ne s'arrêtera pas à cette thèse car d'un goût prononcé pour les mathématiques, tu as fait naître un véritable désir de recherche. Ton enthousiasme, ta clarté, ton optimisme et ton humour ont éclairé cette thèse, je t'en remercie infiniment.

En deuxième année, ma thèse s'intitulait "Apprentissage divers" du nom du projet sur lequel j'ai eu la chance de travailler avec toi, Francis. Cette diversité s'est traduite par la variété des sujets que j'ai pu abordé grâce à ta vision s'étendant de la puissante abstraction à l'application concrète. Ta bienveillance, ta précision, ta présence ont largement contribué à l'accomplissement de cette thèse, comme elles font de cette équipe un environnement exceptionnel de recherche, je t'en remercie beaucoup.

En cette fin de thèse, Zaid, tu m'offres la possibilité de poursuivre l'élan que m'ont offert ces 3 ans. Je t'en remercie et je suis impatient d'entamer de nouveaux travaux avec toi.

Je tiens à remercier Jérôme Malick et François Glineur d'avoir rapporté ma thèse, j'espère profiter au mieux de vos commentaires. Je remercie également Francis Bach, Jérôme Bolte, Zaid Harchaoui d'avoir accepté d'assister à ma soutenance et d'enrichir ces travaux par de plus amples discussions.

Les collaborations qu'ont amené cette thèse furent source de belles rencontres. Fajwel, ce fut un plaisir de partager un bureau, ta bonne humeur ainsi que ta vision pragmatique des problèmes, j'espère que nous aurons l'occasion de nous retrouver dans le futur. Nicolas, ton humour, ta clarté tant orale qu'écrite, ta gentillesse, furent l'une des plus belles découvertes de cette thèse et j'espère à nouveau partager quelques moments avec toi aux Houches par exemple. Je te souhaite la réussite d'une carrière qui offrira à de nombreux étudiants la supervision d'un directeur de thèse exceptionnel. Damien, travailler avec toi fut l'une des plus grandes sources d'enrichissement, la spontanéité de tes idées, ton intuition mathématique m'ont autant étonné que rafraîchit ma vision des problèmes. Je ne doute pas du succès à venir dans tes travaux et j'espère toi aussi te recroiser bientôt.

En 7 ans, nous avons partagé couloir, toit, bureau et manteau ; Jean-Baptiste, tu as toujours été présent tant pour honorer ton surnom, qu'apporter un soutien inestimable dans les épreuves. J'ai peu de mots pour t'exprimer ma gratitude, merci, merci beaucoup. J'espère un jour repartir à la conquête d'un sommet en ta compagnie, je ferais des efforts pour t'attendre cette fois !

Dès ton arrivée en stage, Antoine, j'ai vu une bouille des plus sympathiques qu'il

m'ait été donné de rencontrer. Ta compagnie pendant 2 ans furent une source de joie et de rire qui ont éclairé cette thèse comme je n'aurais pas imaginé en avoir la chance. J'espère bien te retrouver un jour, quelque soit le continent, afin de poursuivre l'étude des bars à chat du monde.

Au-delà des découvertes scientifiques, cette thèse fut l'occasion de rencontres mémorables et précieuses. Merci au prédécesseurs, Guillaume, Piotr, Vincent, Sesh, Loïc, Florent, Rémi, pour leur accueil et leurs conseils avisés, vous étiez l'une des premières belles découvertes de cette thèse. Merci aux postdocs et permanents, Alessandro, Adrien, Robert, Fabian, Igor, Pierre, Rélja, Lénaïc, Anton, Pascal, Simon, pour nous avoir fait partagé leur savoir et leur enthousiasme. Aux compagnons de fortune, Aymeric, Damien, Nicolas, Théophile, Rémi, Gauthier, Julia, Matthew, Guilhem, Christophe, Nastia, Maxime, Dmitri, Tatiana, Gül, Vadim, Alexandre ;votre compagnie durant ces trois ans a fait la richesse de cette thèse tant d'un point de vue scientifique, qu'humoristique ou philosophique. Merci à vous, j'espère bien que nos chemins se recroiseront. Bonne chance aux nouveaux, Loucas, Raphaël, Thomas K, Thomas E, Dmitri, Yana, Antoine, Ignacio, c'était un plaisir de passer quelques temps avec vous, hâte de vous revoir vous aussi.

En cette fin de thèse, j'ai une pensée émue pour Prototo qui n'est jamais passé à l'âge adulte. Marion, Rémi, Thomas, former équipe avec vous était l'une des meilleures expériences de ces trois années. Merci pour l'aventure !

La fin de cette thèse marque aussi celle d'une colocation aussi spacieuse, qu'heureuse. Marion, Marine, Thomas, Anne , merci de m'avoir supporté pendant ces trois ans et merci pour tous les soirs où nous avons eu l'occasion de partager quelques moments dans un foyer.

Depuis dix ans j'ai la chance de partager un peu ma vie avec vous, Josselin, Nathan, Jérémy, Maxime, Nicolas, Lou, Nathan, Aymeric, Éric, Virginie et les autres merci pour votre soutien et d'avoir la joie de vous connaître !

Á mon parrain Marc, pour son soutien et sa présence tout au long des années. Á ma grand-mère, Françoise, avec qui j'ai eu la joie de m'exercer à la rédaction. Á mon grand-père Roger ; je ne saurais encore dire si les mathématiques préexistent à l'univers mais le plaisir d'y jouer est né en moi grâce à toi. Á mes soeurs, Amélie et Hélène, pour leur soutien et la joie des moments passés ensemble. Á mon père dont le regard souriant ne disparaîtra jamais. Á ma mère dont la générosité porte mes pas.

Merci à Damien, Antoine, Thomas, pour les précieuses relectures de ce manuscrit. Merci aux mêmes personnes ainsi qu'à Loucas et Raphaël pour la préparation finale. Enfin merci à Maxime et Nathan pour tous les petits trous !

Contents

Extended abstract

This thesis studies how optimization procedures can take advantage of the geometry of a problem around its solutions in a first part and how they can handle combinatorial structures in a second part. Its outline is the following.

Part 1

The first part analyzes sharpness of functions around their minimizers by so-called error bounds, first discovered by Łojasiewicz.

Chapter 1: This chapter sets up the framework of optimization studied in the first part. Starting from a brief introduction to convex optimization, it further presents Łojasiewicz inequality that guarantees convergence of restart schemes developed in this part. We also briefly present an interpretation of accelerated algorithms that we restart.

Chapter 2: This chapter presents restart schemes of classical algorithms for convex optimization. They provide optimal rates of convergence under generic error bounds on the minimum of the function. Precisely, we study convex optimization problems of the form

$$\text{minimize } f(x)$$

in variable $x \in \mathbb{R}^d$, where f is a convex function.

Problem setting. To tackle both smooth and non-smooth problems, we assume that there exist $1 \leq s \leq 2$ and $L > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|^{s-1}, \quad \text{for all } x, y \in \mathbb{R}^d, \tag{1}$$

where $\nabla f(x)$ is any sub-gradient of f at x. This encompasses smooth convex optimization for $s = 2$ and classical assumption for non-smooth convex optimization for $s = 1$. The optimal rate of convergence for such functions is bounded as $O(1/N^\rho)$, where N is the total number of iterations and

$$\rho = 3s/2 - 1,$$

which gives $\rho = 2$ for smooth functions and $\rho = 1/2$ for non-smooth functions.

Furthermore, we assume that f satisfies a Hölderian error bound on a compact K that contains the set of minimizers X^* of the problem, that is, there exist $r \geq 1$, and $\mu > 0$ such that

$$\mu d(x, X^*)^r \leq f(x) - f^* \quad \text{for all } x \in K \tag{2}$$

where f^* is the minimum of f and $d(x, X^*)$ is the distance from x to X^*, the set of minimizers of the problem. This encompasses sharp functions for $r = 1$, strongly convex functions for $r = 2$ and it is known to be satisfied for a broad class of functions as the Łojasiewicz inequality.

The smoothness assumption (1) defines an upper bound on the function. Combined with the previous lower bound, we observe that necessary $s \leq r$. The convergence rate of our restart schemes will then depend on the following condition number, based on exponents s ad r,

$$\tau = 1 - s/r \in [0, 1[.$$

Optimal restart schemes. Under these assumptions, we restart classical algorithms for convex optimization, namely Nesterov's accelerated algorithm if the function is smooth ($s = 2$) or a fast universal gradient algorithm for general convex functions satisfying equation (1). Precisely, we run these algorithms until some number of iterations scheduled in advance, stop them and restart them from the last iterate. If all parameters of the function are known, we retrieve optimal rates of convergence for the class of functions satisfying our hypotheses, i.e., the precision reached at the last point $\hat{x}$ is upper bounded as

$$f(\hat{x}) - f^* \;=\; O\left(\exp(-\kappa^{-\frac{s}{2\rho}} N)\right), \quad \text{when } \tau = 0,$$

while,

$$f(\hat{x}) - f^* \;=\; O\left(\kappa^{\frac{s}{2\tau}} N^{-\frac{\rho}{\tau}}\right), \quad \text{when } \tau > 0,$$

where N is the total number of iterations and κ is a generalized condition number that matches the classical one for smooth and strongly convex functions. The error bound assumption, incorporated in the parameter τ, therefore results in faster rates than the ones obtained with only the smoothness hypothesis. Our proof is simple and the use of universal gradient algorithms enables to cover the non-smooth case.

Adaptive restart schemes. We then develop an adaptive restart scheme if the function is smooth ($s = 2$) and if the parameters μ, r of the error bound (2) are unknown. Namely, for a fixed budget of iterations, a log-scale grid-search on parameters μ, r is proven to be nearly optimal up to an additional cost of $\log(N)$. We also develop restart schemes using a stopping criterion based on the gap $f(x) - f^*$, when the optimal value f^* is known in advance. These methods do not need any parameters of the function and are proven to be optimal.

Extensions. Finally, we extend our results for composite problems in non-Euclidean settings. This enables to treat constrained or ℓ_1 regularized problems for which our restarts are shown empirically to outperform plain implementations of

accelerated algorithms.

Chapter 3: This chapter presents sparse problems that originally attempt to decode a vector with few non-zero coordinates given some linear observations of it. Dedicated optimization procedures and recovery problems of group sparse vectors or low rank matrices are discussed.

Chapter 4: This chapter studies how optimization complexity and recovery performance of sparse problems are related. Namely, we study recovery problems of a sparse vector $x^* \in \mathbb{R}^n$ given p linear observations $b_i \approx a_i^T x^*$ for $i = 1, \dots, n$. If observations are exact, x^* is decoded by the exact recovery problem

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & Ax = b. \end{array} \tag{3}$$

in $x \in \mathbb{R}^n$. Otherwise, the robust recovery problems reads

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & \|Ax - b\|_2 \leq \epsilon, \end{array} \tag{4}$$

in $x \in \mathbb{R}^n$, where ϵ is a tolerance on the noise in the observations. The performance of these decoding procedures to recover x^* has been extensively studied in the compressed sensing literature, either to know how many observations are needed to retrieve x^* in the exact case or how sensitive is the robust problem to noise. Here, we relate these statistical measures of performance to the computational complexities of these problems.

Exact recovery. For problem (3), we show that recovery of x^* is equivalent to the sharpness of the problem around its minimizers, which reads

$$\gamma \|x - x^*\|_1 < \|x\|_1 - \|x^*\|_1, \quad \text{for all } x \text{ s.t. } Ax = b$$

where $0 \leq \gamma < 1$. We develop then linearly convergent restart schemes of the smoothing technique of Nesterov, whose rate is controlled by the sharpness constant γ. A statistical analysis reveals that γ is then controlled by the recovery threshold of the problem, i.e., the maximal sparsity level of x^* such that it can be recovered. Overall, this shows that the more performing the decoding procedure is, the easier the decoding problem is.

Robust recovery. The sensitivity to noise of the robust recovery problem is well-known to be measured by conically restricted singular values of the coding matrix A. We show that these latter control the sharpness of the exact recovery problem, hence linear rate of our restart scheme, but also the complexity of oracle based techniques such as the ellipsoid method. Overall, we identify a single quantity that controls both the recovery performance and the computational complexity of decoding procedures. This is then further illustrated by numerical experiments.

Generalized sparse structure. Our analysis is extended to generalized recovery problems of either group sparse vectors or low rank matrices. We identify same sharpness property of the corresponding decoding objective. We then highlight their conic nature and define appropriate conically restricted singular values to generalize our results.

Part 2

This part studies machine learning procedures that simultaneously solve a prediction problem and simplify it by grouping either features, samples or tasks. The approach mixes classical empirical loss minimization procedures and partitioning problems. The resulting models are non-convex but we provide systematic algorithmic strategies, whose performances are illustrated on either real or synthetic data.

Chapter 5: This chapter introduces supervised machine learning problems and the interest of grouping features, samples or tasks in the context of big data. A general overview of the approach is presented, followed by key tools for partitions or non-convex optimization problems.

Chapter 6: This chapter studies the problem of grouping features while solving a prediction task like regression or classification. The task is formulated as a classical loss minimization problem with additional partitioning constraints on the features, that reads, for regression,

$$\begin{array}{ll} \text{minimize} & L(w) + R(w) \\ \text{subject to} & w = Zv, \quad Z \in \{0,1\}^{d \times Q}, \quad Z\mathbf{1} = \mathbf{1} \end{array} \tag{5}$$

in variables $w, v \in \mathbb{R}^d$ and Z, where L and R are respectively the empirical loss and the regularization of the prediction problem and $Z \in \{0,1\}^{d \times Q}$ is the assignment matrix of the features in Q groups, within each all features share a same weight given in the vector v.

Optimization strategies. For the squared loss, analytic minimization in variables w and v enables to isolate the partitioning problem in terms of normalized equivalence matrices of partitions that encode if pairs of points belong to the same group. A convex relaxation on the convex hull of the set of normalized equivalence matrices can then be performed by Frank-Wolfe algorithm, whose linear minimization oracle amounts to a k-means problem that can be solved exactly for regression.

Although feasible set of (5) is non-convex, projection on it amounts to a k-means problem that can once again be solved exactly for regression. A projected gradient descent scheme can then be applied, which offers a scalable algorithmic approach for any loss.

Theoretical analysis. The feasible set of (5) is a union of subspaces defined by partitions. The projected gradient scheme can therefore be analyzed by analogy with the Iterative Hard Thresholding algorithm used for sparse problems. Its convergence

depends on a restricted isometric property of the problem along the subspaces defined by partitions. While for sparsity problem this condition is satisfied for a number of samples that is only a fraction of the number of features, the problem of grouping features requires as many samples as features to ensure convergence. The underlying combinatorial problem appears too complex to be solved with few samples by a projected gradient scheme.

Extension to sparse and grouped vectors. Finally a projected gradient scheme to both select and group variables is presented which enjoys the theoretical guarantees of sparse problems while highly reducing the dimensionality of the problem by grouping features.

Chapter 7: This chapter studies the problem of grouping samples while solving a prediction task. It can either be seen as a supervised clustering task or a prediction problem allowed to output diverse predictions. As for features, the problem is formulated as a supervised learning problem constrained by partitions on the data. Same strategies are used to solve it: a convex relaxation using Frank-Wolfe and a projected gradient scheme, both requiring the solution of a k-means problem to perform an iterative resolution of the problem.

Chapter 8: This chapter studies the problem of grouping tasks for classification. The underlying partitioning problem is isolated, which enables the development of a projected gradient scheme for general losses. In the special case of a squared loss, we show that the problem reduces to a k-means problem.

References

The publications related to this manuscript are listed below:

a) Interpretation of Accelerated algorithm in Chapter 1, Section 1.4 is based on the article: Integration Methods and Accelerated Optimization Algorithms, D. Scieur, V. Roulet, F. Bach and A. d'Aspremont to appear in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*.

b) Chapter 2 is based on the article: Sharpness, Restart and Acceleration, V. Roulet and A. d'Aspremont, to appear in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*.

c) Chapter 4 is based on the article: Computational Complexity versus Statistical Performance on Sparse Recovery Problems, V. Roulet, N. Boumal and A. d'Aspremont, under submission to *Information and Inference: A Journal of the IMA*.

d) Chapters 6, 7, 8 are based on the article: Supervised Learning in the Data Cube, V. Roulet, F. Fogel, F. Bach and A. d'Aspremont, presented at workshop *Transfer and Multi-Task Learning: Trends and New Perspectives (NIPS 2015)* and the preprint: Learning with Clustering Penalties, V. Roulet, F. Fogel, F. Bach and A. d'Aspremont, *ArXiv 1506.04908*

e) Chapter 6 is also based on the article: Iterative Hard Clustering of Features, V. Roulet, F. Fogel, F. Bach and A. d'Aspremont, under submission to the *21st International Conference on Artificial Intelligence and Statistics (AISTATS 2018)*

Part I

Convex optimization with error bounds

Chapter 1

Introduction

Optimization problems consist in finding extremal values of functions, such as the minimal cost of a task, and take the general form

$$\text{minimize } f(x) \tag{1.1}$$

in variable $x \in \mathbb{R}^d$, where f, the *objective function*, has at least one minimizer. Analytical solutions of such problems are generally not available. Iterative algorithmic procedures are therefore developed to get an approximate solution $\hat{x}$ of (1.1). The *precision* achieved by a method is measured by the gap $f(\hat{x}) - f^*$ between the estimate solution $f(\hat{x})$ and the true solution $f^* = \min_x f(x)$. The *complexity* of an algorithm to solve (1.1) is then defined as the number of iterations needed to achieve an accuracy ε.

Development of algorithms to solve the optimization problem depend on the information available at each iteration. Throughout this thesis we are interested in *first order* algorithms that have access to gradient (or sub-gradients defined below) $\nabla f(x)$ of the function at any querying point $x \in \mathbf{dom}\, f$. When dimension d is large, their cheap iteration cost makes them more appropriate than second order methods that require to compute the Hessian of the function.

Assumptions on the function can then be exploited to build appropriate methods. In this introduction we recall how convexity and smoothness were used to develop efficient algorithms. We then present Łojasiewicz inequality that merely describes the behavior of the objective function around its minimizers and show how it can be exploited by restart schemes. Finally, we briefly present how accelerated gradient algorithms for convex functions can be interpreted as discretization methods of the gradient flow.

1.1 Convex optimization

Convexity has quickly raised the attention of researchers as a key property that enables efficient resolution of optimization problems. Several books [Boyd and Vandenberghe, 2004; Bertsekas, 1999; Borwein and Lewis, 2010; Nocedal and Wright, 1999; Nesterov, 2013*b*] provide exhaustive presentation of the subject. Here we briefly

present the key assumptions and classical algorithms to highlight the interest of characterizing the geometry of the problem around its minimizers.

1.1.1 Convexity

We first recall elementary definitions. A convex set, as defined below, is a set that contains any segment of its points.

Definition 1.1.1. ***Convex sets*** *A set Q in $\mathbb{R}^d$ is convex if for any $x, y \in Q$ and $\theta \in [0, 1]$,*

$$\theta x + (1 - \theta)y \in Q.$$

Convexity of a function f can then be defined by the convexity of its epigraph $\mathcal{E}(f) = \{(x, t) \in \mathbb{R}^d \times \mathbb{R} : f(x) \leq t\}$[Rockafellar, 2015] or equivalently as follows.

Definition 1.1.2. ***Convex functions 0^{th} order*** *A function f is convex if its domain is convex and if for any $x, y \in \mathbf{dom}\, f$ and $\theta \in [0, 1]$,*

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y).$$

If the function is in addition differentiable, then it is convex if it is lower bounded at any point in its domain by its linear approximation. It is detailed in following equivalent definition.

Definition 1.1.3. ***Convex functions 1^{st} order*** *A differentiable function f is convex if its domain is convex and if for any $x \in \mathbf{dom}\, f$*

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle, \quad \textit{for every } y \in \mathbf{dom}\, f.$$

This property is generalized for non-differentiable convex functions through the notion of sub-gradients defined below.

Definition 1.1.4. ***Sub-gradient*** *Let f be a convex function. A vector g is called the sub-gradient of f at point $x \in \mathbf{dom}\, f$ if*

$$f(y) \geq f(x) + \langle g, y - x \rangle, \quad \textit{for every } y \in \mathbf{dom}\, f.$$

Closed [1] convex functions have a sub-gradient at any point in the interior of their domain [Nesterov, 2013*b*]. In the following, we denote $\nabla f(x)$ any sub-gradient of a closed convex function f at a given point x. Convexity offers a simple certificate of optimality : if $x \in \mathbb{R}^d$ possesses a null (sub)gradient $\nabla f(x) = 0$ then it is a global minimizer. While convexity offers affine lower bounds at each point, quadratic lower bounds can be obtained through the notion of strong convexity as detailed in next section.

1. A function is closed if its epigraph is closed, or equivalently if it is lower-semi-continuous

1.1.2 Strong convexity

We first recall general definition of strong convexity.

Definition 1.1.5. Strong convexity 0^{th} order *A function f is strongly convex if its domain is convex and if there exists $\mu \geq 0$ such that for any points $x, y \in \mathbf{dom}\, f$ and $\theta \in [0,1]$,*

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y) - \theta(1-\theta)\frac{\mu}{2}\|x-y\|_2^2$$

It follows that strong convexity implies convexity. Besides when the function is differentiable, following equivalent definition details lower bound induced by strong convexity.

Definition 1.1.6. Strong convexity 1^{st} order *A differentiable function f is strongly convex if its domain is convex and if there exists $\mu \geq 0$ such that for any points $x \in \mathbf{dom}\, f$,*

$$f(y) \geq f(x) + \langle \nabla f(x), y-x \rangle + \frac{\mu}{2}\|x-y\|_2^2, \quad \textit{for every } y \in \mathbf{dom}\, f.$$

In general closed convex functions can also be lower bounded by quadratics in the interior of their domain, several lower bounds may exist at one point defined by the different sub-gradients of the function. Strong convexity can be refined to analyze monomial lower bounds on the function through the notion of uniform convexity. We simply give its definition for differentiable functions to highlight the resulting lower bound and refer to e.g. Juditski and Nesterov [2014]; Bauschke and Combettes [2011] for further details.

Definition 1.1.7. Uniform convexity *A differentiable function f is uniformly convex if its domain is convex and if there exists $r \geq 2$, $\mu \geq 0$ such that for any points $x \in \mathbf{dom}\, f$,*

$$f(y) \geq f(x) + \langle \nabla f(x), y-x \rangle + \frac{\mu}{2}\|x-y\|_2^r, \quad \textit{for every } y \in \mathbf{dom}\, f.$$

As it will be shown in Section 1.1.4, strong convexity or uniform convexity enables much more efficient resolution of optimization problems than plain convexity. However it is a strong assumption as it requires quadratic lower bounds at any point and any direction. We present in Section 1.2, how it can be relaxed by much weaker assumption on the function while still getting fast rates of convergence.

1.1.3 Smoothness

A standard way to design algorithms for (1.1) is to minimize an upper bound on the objective function f at each iteration. These can be derived from its Taylor

expansion at a point $x \in \mathbf{dom}\, f$ that reads for a differentiable function f,

$$\begin{aligned} f(y) &= f(x) + \int_0^1 \langle \nabla f(x + \tau(y - x)), y - x \rangle d\tau \\ &= f(x) + \langle \nabla f(x), y - x \rangle + \int_0^1 \langle \nabla f(x + \tau(y - x)) - \nabla f(x), y - x \rangle d\tau \end{aligned}$$

Bounds on the gradient lead then to upper bounds on the function. In this section we use Euclidean norm to define smoothness, refined assumptions are presented in Section 1.1.6.

Smooth functions

The most common assumption for differentiable function is that their gradient is Lipschitz continuous as defined below.

Definition 1.1.8. ***Smooth functions*** *A differentiable function f is smooth if there exists $L > 0$ such that*

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2, \quad \textit{for every } x, y \in \mathbf{dom}\, f.$$

At a given point $x \in \mathbf{dom}\, f$, using the Taylor expansion of f, smoothness implies

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|x - y\|_2^2, \quad \text{for every } y \in \mathbf{dom}\, f.$$

Denoting $x_+ = x - \frac{1}{L}\nabla f(x)$ the minimizer of this upper bound, we get

$$f(x_+) - f(x) \leq -\frac{1}{2L}\|\nabla f(x)\|_2^2,$$

that ensures decreasing of the objective if the current point is not a stationary point ($\nabla f(x) = 0$). This is then used to derive convergence rates of the gradient descent for convex functions [Nesterov, 2013*b*] or gradient dominated functions [Karimi et al., 2016].

Non-smooth functions

If the function is not differentiable, Taylor expansions cannot be used. However, if the function is closed convex, bound on its sub-gradients still offer local upper bounds on the function. To this end we make the following assumption.

Assumption 1.1.9. ***Non-smooth functions*** *For a non-differentiable closed convex function f, there exists $L > 0$ such that*

$$\|\nabla f(x)\|_2 \leq \frac{L}{2}, \quad \textit{for every } x \in \mathbf{dom}\, f,$$

where $\nabla f(x)$ is any sub-gradient of f at x.

At a given point $x \in \mathbf{dom}\, f$, with a given sub-gradient $\nabla f(x)$, this ensures

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + L\|x - y\|_2, \quad \text{for every } y \in \mathbf{dom}\, f.$$

Minimization of this upper bound may not be tractable but approximate minimizations can exist [Devolder et al., 2014; Nesterov, 2015]. Notice that non-smooth convex optimization methods such as sub-gradient [Nesterov, 2013*b*], dual averaging [Nesterov, 2009], double dual averaging [Nesterov and Shikhman, 2015], generally rely cutting planes arguments. Here we focus on upper bounds to derive generic bounds that encompass both smooth and non-smooth cases.

Hölder-smooth functions

Smoothness of a differentiable function f can be refined by considering Hölder Lipschitzity of its gradient. While difficult to observe, it allows a common treatment of the smooth and non-smooth cases by the following definition.

Definition 1.1.10. ***Hölder smooth functions*** *A closed convex function f is Hölder smooth if there exist $1 \leq s \leq 2$, $L > 0$ such that*

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2^{s-1}, \quad \textit{for every } x, y \in \mathbf{dom}\, f,$$

where $\nabla f(x), \nabla f(y)$ are any sub-gradient of f at respectively x and y.

Notice that if $s > 1$, the function is necessarily differentiable, since the set of sub-gradients at any point is then reduced to a singleton. If $s = 2$, we retrieve the definition of smooth functions, if $s = 1$ the assumption on non-smooth convex functions and for general $1 \leq s \leq 2$ we get a refined upper bound on the function at a given point $x \in \mathbf{dom}\, f$ that reads

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{s}\|x - y\|_2^s, \quad \text{for every } y \in Q. \tag{1.2}$$

Even though exact minimization of such upper bounds might not be tractable, approximate minimizations [Devolder et al., 2014; Nesterov, 2015] are possible and lead to universal gradient methods presented in next section.

1.1.4 Classical algorithms for convex optimization

In this section we present classical algorithms to solve unconstrained convex optimization problems that read

$$\text{minimize } f(x) \tag{1.3}$$

in variable $x \in \mathbb{R}^d$, where f is a closed convex function. We detail their complexity depending on additional assumptions on the function such as smoothness or strong convexity. For each class of problems, Nemirovskii and Yudin [1983] computed the least number of iterations an algorithm must build to achieve a given precision for any function in the class. These define lower complexity bounds of convex optimization

problems. Algorithms that achieve these lower complexity bounds are then called optimal.

In the following we denote x_0 the starting point of the algorithms and $d(x_0, X^*) = \min_{y \in X^*} \|x_0 - y\|_2$ the Euclidean distance between the starting point x_0 and the set of solutions $X^* = \operatorname{argmin}_{y \in Q} f(y)$. Detailed implementations for general convex problems are presented in Appendix A.

Gradient descent

Gradient descent dates back from Cauchy [1847]. It simply moves along the opposite direction of the gradient at each iteration to progressively decrease the objective values as

$$x_{t+1} = x_t - h_t \nabla f(x_t),$$

where x_t is the current iterate, x_{t+1} is the next iterate and h_t is the step size. Several choices exist for the step-size depending on the assumptions made on the function.

• **Smooth convex functions** If the function f is L-smooth, a constant step-size $h_t = \frac{1}{L}$ ensures convergence. Gradient descent outputs then after t iterations a point $\hat{x}$ that satisfies

$$f(\hat{x}) - f^* \leq \frac{L}{t} d(x_0, X^*)^2$$

If L is not known in advance, a backtracking line-search as presented by Nesterov [2013*a*] estimates it on the fly to get the same rate with an additional log factor cost.

• **Smooth strongly convex functions** If the function f is L-smooth and μ-strongly convex, constant step size still ensures convergence, however a bigger step-size can be used, namely $h_t = \frac{2}{\mu + L}$. Gradient descent shows then linear convergence, it outputs after t iterations a point $\hat{x}$ that satisfies

$$f(\hat{x}) - f^* \leq \left(\frac{1 - \mu/L}{1 + \mu/L} \right)^t (f(x_0) - f^*).$$

Notice that if μ is not known in advance, constant step size $h_t = \frac{1}{L}$ still ensures linear convergence as

$$f(\hat{x}) - f^* \leq \left(1 - \frac{\mu}{L} \right)^t (f(x_0) - f^*).$$

Although worse than previous one, this complexity bound illustrates that gradient descent adapts to the assumptions on the problem. If the function is strongly convex it automatically shows linear rate. In fact the function only needs to satisfy gradient dominated property presented later to get linear rate. However since gradient descent is not optimal, more elaborate schemes can get better linear rates.

• **Non-smooth settings** We briefly discuss classical algorithms for non-smooth optimization. They can be tackled by universal gradient algorithms or restart schemes

presented later. If the function is non-smooth, sub-gradients can be used instead of gradients leading e.g. to sub-gradient methods [Nesterov, 2013*b*], dual averaging method [Nesterov, 2009] or double dual averaging methods [Nesterov and Shikhman, 2015], unified recently by Ito and Fukuda [2016]. They all output after t iterations a point t that satisfies the optimal rate for this class of function, i.e.

$$f(\hat{x}) - f^* = O(1/t^{1/2}).$$

However they all need to know in advance an additional parameter such as a bound $R \geq d(x_0, X^*)$ on the distance to the set of minimizers, which can be simply the size of the constrained set for constrained optimization presented later.

If the function is non-smooth but strongly convex, sub-gradient methods as presented e.g. by Lacoste-Julien et al. [2012] can achieve after t iterations the optimal rate in this case, i.e.

$$f(\hat{x}) - f^* = O(1/t).$$

Accelerated gradient descent

• **Smooth convex functions** In his seminal work Nesterov [1983] developed an accelerated scheme for the optimization of $L-$smooth convex functions f. It was originally motivated by the construction of a lower bound on the function using its convexity. It has been revisited several times by the author himself [Nesterov, 2013*a*, 2005, 2015] or by others [Tseng, 2008] to cite a few. Overall, even if implementation may vary among the authors, accelerated gradient algorithm outputs after t iterations a point $\hat{x}$ that satisfies

$$f(\hat{x}) - f^* \leq \frac{4L}{t^2} d(x_0, X^*)^2$$

It achieves then optimal rate of convergence for the class of smooth convex functions Here again the smoothness constant L can be estimated on the fly [Nesterov, 2013*a*], such that the algorithm does not need to know any parameter of the function to be run.

• **Smooth strongly convex functions** Nesterov [2013*b*] presented also acceleration algorithms for L-smooth μ-strongly convex functions f, which outputs after t iterations a point $\hat{x}$ that satisfies the optimal rate of convergence, precisely

$$f(\hat{x}) - f^* \leq \left(1 - \sqrt{\frac{\mu}{L}}\right)^t (f(x_0) - f^*).$$

As ratio μ/L is often small, getting a $\sqrt{\mu/L}$ rate has important impact on the convergence. However while gradient descent automatically adapts to strong convexity, implementation of the accelerated algorithm varies considerably for strongly convex functions. Strong convexity parameter μ can be estimated by doing outer loops of the algorithm as mentioned by Nesterov [2013*a*] and further developed by Fercoq and Qu [2017].

Notice that for quadratic functions defined by positive definite matrices, accelerated algorithm of Nesterov [2013*b*] is not exactly optimal. Conjugate gradients [Hestenes and Stiefel, 1952] or Heavy ball method [Polyak, 1964] achieve a slightly better rate. However these are not proven to converge for general smooth and strongly convex functions [Lessard et al., 2016].

Universal gradient methods

Hölder smoothness was first tackled by Nemirovskii and Nesterov [1985] when parameters s and L of the function were known. Recently Nesterov [2015] presented universal methods able to get optimal rates on this class of function without knowing parameters s and L in advance but only the desired target precision. A so-called universal gradient method was developed that simply approximates the minimization of the upper bound (1.2). An accelerated version incorporates then the convexity assumption similarly as in the classical smooth case. For a Hölder smooth function with parameters (s, L) and required accuracy ε, it starts from a feasible point x_0 and outputs after t iterations a point $\hat{x}$ that satisfies

$$f(x) - f^* \leq \frac{\varepsilon}{2} + \frac{c_s L^{\frac{2}{s}} d(x_0, X^*)}{\varepsilon^{\frac{2}{s}} t^{\frac{3s-2}{s}}} \frac{\varepsilon}{2},$$

where $c_s = 2^{4-2/s}$. If the smoothness parameters s, L are known, target accuracy ε can be optimized which leads to optimal rates for the class of Hölder smooth convex functions [Nemirovskii and Nesterov, 1985], i.e. after t iterations,

$$f(x) - f^* \leq O(1/t^{3s/2-1}).$$

In particular if the function is smooth ($s = 2$) optimal target accuracy is $\varepsilon = 0$ and we retrieve optimal rate for smooth convex functions $O(1/t^2)$. For non-smooth functions with bounded sub-gradients, we also get the optimal rate $O(1/t^{1/2})$.

Other algorithms

Although not studied here, quasi-Newton methods such as BFGS Broyden [1970]; Fletcher [1970]; Goldfarb [1970]; Shanno [1970] that approximate the Hessian along the iterations show fast numerical convergence. Their theoretical analysis is however more complex. Recently Scieur et al. [2016] proposed also regularized polynomial extrapolation methods that shows good numerical performance. Finally non-smooth uniform convex functions were tackled by restart schemes by Juditski and Nesterov [2014] that lead to optimal rate.

1.1.5 Constrained and composite problems

Previous problems were presented for convex unconstrained problems of the form (1.3) for sake of clarity. We detail here how constrained or regularized problems are treated.

Constrained convex optimization problems read

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in Q, \end{array} \tag{1.4}$$

in variable $x \in \mathbb{R}^d$, where f is closed convex and $Q \subset \mathbb{R}^d$ [2], that encodes *constraints* on variables, is a closed convex set. For such problems we assume that we can compute feasible points for problem (1.1) from any $x \in \mathbb{R}^d$. This generally amounts to have access to the *Euclidean projection* on Q but refined assumptions exist such as linear minimization [Frank and Wolfe, 1956; Jaggi, 2013] or proximal operation on the set for mirror gradient descent algorithms [Beck and Teboulle, 2003]. Additional assumptions on the function such as smoothness or strong convexity presented in Sections 1.1.3 and 1.1.2 need only to be valid on the set of constraints. Parameters L and μ of these assumptions can then be much better when restricted to the set Q. Algorithms can then take advantage of the specific geometry of the function on the set as presented by d'Aspremont et al. [2013].

Constrained convex optimization problems belong to the more general class of composite problems that read

$$\text{minimize } f(x) + g(x) \tag{1.5}$$

in variable $x \in \mathbb{R}^d$, where f is closed convex and g is a "simple" convex function in the sense that its proximal operator

$$\mathbf{prox}_g^\gamma(x) = \operatorname*{argmin}_{y \in \mathbb{R}^d} g(y) + \frac{\gamma}{2}\|x - y\|_2^2$$

can be solved at a cheap computational cost for any $x \in \mathbb{R}^d$ and $\gamma \geq 0$. Constrained problems can be cast as composite problems by taking g the indicator function of the set Q whose proximal operator is the Euclidean projection on Q. But this encompasses also least squares problems regularized by sparsity inducing norms [Bach et al., 2012], such as the ℓ_1 norm.

Algorithms presented in Section 1.1.4 can be adapted to incorporate the "simple" function g [Nesterov, 2013*a*; Beck and Teboulle, 2009; Tseng, 2008]. They achieve same rates of convergence for the classes of functions that f belongs to. However, as already mentioned, parameters of these assumptions can considerably change. In other words constraining or regularizing the problem may make the problem easier.

1.1.6 Beyond Euclidean geometry

We defined in Sections 1.1.3 and 1.1.2 with respect to the Euclidean norm. However, in some cases, different norms are more appropriate to describe the function on a constrained set Q. To handle them, Bregman divergences of strongly convex functions are used as surrogates of the norm of interest (see e.g. Beck and Teboulle [2003]

2. Without loss of generality we assume $Q \subset \mathbf{dom}\, f$

for a detailed presentation). This leads to the family of mirror gradient [Nemirovskii and Yudin, 1983] or accelerated mirror gradient descents [Krichene et al., 2015] and can been incorporated in numerous algorithms such dual averaging [Nesterov, 2009] or universal gradient algorithms [Nesterov, 2015].

Recently Lu et al. [2016]; Bauschke et al. [2016] introduced smoothness and strong convexity with respect to another convex function that refines even more the description of convex functions.

1.2 The Łojasiewicz inequality

1.2.1 Definitions

In his pioneering work, Łojasiewicz [1958, 1961, 1965] proved two inequalities on semi-algebraic functions. The first one lower bounds the function values by the distance to the set of minimizers. We give a simple definition that will be used in Chapter 2.

Definition 1.2.1. ***Łojasiewicz error bound inequality*** *A lower bounded continuous function f on $\mathbb{R}^d$ with non-empty set of minimizers X^* satisfies the* Łojasiewicz error bound inequality *on a set $K \supset X^*$ if there exists $\mu > 0$, $r \geq 1$ such that*

$$\mu d(x, X^*)^r \leq f(x) - f^*, \quad \text{for every } x \in K, \tag{1.6}$$

where $f^* = \min_{x \in \mathbb{R}^d} f(x)$ and $d(x, X^*) = \min_{y \in X^*} \|x - y\|_2$ is the Euclidean distance between x and the set of minimizers of f.

The first result on error bounds dates back from Hoffman [1952] who analyzed systems of linear inequalities. It was then studied for convex functions by Robinson [1975]; Mangasarian [1985]; Auslender and Crouzeix [1988]. Łojasiewicz [1958, 1961, 1965] extended these results. As it merely describes function in the neighborhood of its minimizers, Łojasiewicz error bound is generally satisfied. Its proof for general subanalytic functions can be found for example in Bierstone and Milman [1988, Theorem 6.4] using topological arguments. Error bounds have been more generally expressed using residual function of the set of minimizers, see e.g. Pang [1997] for a review. Quantitative results were developed by Luo and Luo [1994]; Luo and Pang [1994]; Luo and Sturm [2000]; Dedieu [1992] and recently further analyzed by Li [2013]; Li, Mordukhovich and Pham [2015]; Li, Mordukhovich, Nghia and Pham [2015]; Beck and Shtern [2015]; Vui [2013].

Although Łojasiewicz inequality is widely applicable, pathological behavior appears when the function is extremely flat or present wild oscillations around their minimizers such as $f(x) = \exp(-1/x^2)$ or $f(x) = \sin(1/x)\exp(-1/x^2)$ extended by continuity on 0. In the presented definition, the larger is r, the flatter is the function around its minimizers and so the worse it is conditioned as presented in Chapter 2. Notice finally that Łojasiewicz error bound inequality is a local assumption that may not generalize to the whole domain of f. Yet, if this property is valid on an open set

$K \supset X^*$, it will also be valid on any compact set $K' \supset K$ with the same exponent r but with a potentially lower constant μ.

The second Łojasiewicz inequality upper bounds gradient magnitude by function values. Again, we give a simple definition that can be generalized.

Definition 1.2.2. ***Łojasiewicz gradient inequality*** *A differentiable function f satisfies Łojasiewicz gradient inequality in the neighborhood K of a critical point x^*, if there exists $c > 0$ and $\theta \in [0, 1[$ such that*

$$c\|\nabla f(x)\|_2^\theta \geq f(x) - f(x^*), \quad \textit{for every } x \in K. \tag{1.7}$$

First discovered by Łojasiewicz for analytic functions, above gradient inequality was generalized by Kurdyka [1998] for semi-algebraic functions by introducing desingularizing functions. It has recently been generalized to the non-smooth case by Bolte et al. [2007].

1.2.2 Variants of strong convexity

As strong convexity appears too restrictive for many problems, it was relaxed in several manners using the definitions of essential strong convexity, weak strong convexity or restricted secant inequality [Karimi et al., 2016; Necoara et al., 2015; Liu and Wright, 2015; Zhang, 2017]. Among these assumptions, the quadratic error bound ((1.6) with $r = 2$) is proven to be the weakest [Karimi et al., 2016].

More generally uniform convexity with parameter r can easily be seen to satisfy Łojasiewicz error bound with some exponent by looking at its first order definition. By allowing exponent r to vary, Łojasiewicz error bound can also describes sharp functions ($r = 1$) studied for example by Gilpin et al. [2012]. Combined with Hölder smoothness, these offer a generic description of convex problems.

1.2.3 Applications

Łojasiewicz gradient inequality (1.7) for $\theta = 1/2$ is better known in the optimization community as the gradient dominated property introduced by Polyak [1963]. It suffices to prove linear convergence of convex functions as recalled by Karimi et al. [2016]. Generic error bounds were also considered by Nemirovskii and Nesterov [1985] to derive generic optimal algorithms for convex functions. However these require to know in advance parameters μ and r which are generally hard to estimate outside the strongly convex case.

Recently the growing interest about Łojasiewicz inequalities led to new convergence analysis of many important algorithms in non-smooth and even non-convex settings [Attouch et al., 2013; Bolte et al., 2014; Frankel et al., 2015]. Generic methods to derive convergence rates for convex optimization were notably studied by Bolte et al. [2015]. In Chapter 2, we present simple restart schemes for convex optimization that take advantage of Łojasiewicz error bound and show adaptivity to parameters μ and r for smooth functions.

Łojasiewicz inequalities have also an impact beyond optimization. They notably lead to the characterization of smooth or non-smooth dynamical systems [Simon, 1983; Bolte et al., 2007] or can provide a simple framework for the development of concentration inequalities [Blanchet and Bolte, 2016].

1.3 Restart schemes

A simple way to achieve acceleration for strongly convex problems is simply to restart the accelerated method for smooth convex functions at regular time intervals [Nesterov, 2013*a*]. Consider the unconstrained minimization of a smooth strongly convex functions. At the unique minimizer x^*, strong convexity reads

$$f(x) - f^* \geq \frac{\mu}{2}\|x - x^*\|_2^2$$

After $t = \lceil e\sqrt{8L/\mu} \rceil$ iterations of the accelerated method for smooth convex functions, one obtains a point $\hat{x}$ satisfying

$$f(\hat{x}) - f^* \leq \frac{4L}{t^2}\|x_0 - x^*\|_2^2 \leq \frac{8L}{\mu t^2}(f(x_0) - f^*) \leq e^{-2}(f(x_0) - f^*)$$

Therefore repeating this operation and counting the total number of iterations leads to the optimal complexity for strongly convex functions. Observe that only the Łojasiewicz error bound implied by strong convexity is used to achieve such rate. This remark was further explored by Nemirovskii and Nesterov [1985] for general convex problems.

The scheduling of the restart schemes is their drawback. In the strongly convex case for example, strong convexity parameter μ must be known in advance. Numerous heuristics strategies have thus been developed to restart the accelerated method for smooth convex functions. O'Donoghue and Candes [2015] proposed to enforce monotonicity of the objective values, that is, to restart the algorithm when the objective values increase. Giselsson and Boyd [2014] proved that such heuristics converge at least as fast as the accelerated scheme for smooth convex functions. However in practice restart show much better performance than a plain implementation of the accelerated algorithm, in particular for composite problems such as LASSO [Tibshirani, 1996]. Continuous time interpretations of the accelerated scheme were also used to justify restart heuristics. They rely on an inertia argument of the continuous time dynamic [Su et al., 2014; Wibisono et al., 2016]. However no theoretical guarantees of this heuristic were proven.

Developing algorithms adaptive to unknown strongly convex parameter was first studied by Nesterov [2013*a*]; Lin and Xiao [2014] by running several outer loops of the accelerated algorithm for strongly convex functions. Recently Fercoq and Qu [2016] proposed restart schemes for generic quadratic error bounds ((1.6) with $r = 2$) robust to misspecification of the parameters. Fercoq and Qu [2017] refined their algorithm to provide adaptive restarts with theoretical guarantees under the same assumptions,

which offer guarantees of accelerated linear rates for the LASSO.

Restart schemes were also studied for non-smooth convex optimization. Juditski and Nesterov [2014] restart the dual averaging method [Nesterov, 2009] to optimize non-smooth uniformly convex functions. They present an adaptive algorithm in this case for a fixed number of iterations and with an additional estimate of the distance to the set of minimizers. Gilpin et al. [2012] restart smoothing technique of Nesterov [2005] to solve zero-sum games. Interestingly, as the optimum value of their problem is known, their technique do not need additional parameters. Finally smooth convex optimization problems satisfying (1.7) with $\theta > 1/2$ were studied by Yang [2016], where they use magnitude of the gradient to provide criteria for restarts.

1.4 Interpretation of accelerated algorithm

Nesterov's accelerated gradient algorithm [Nesterov, 2013*b*] was designed with optimal complexity in mind, but the proof relies on purely algebraic arguments and the key mechanism behind acceleration remains elusive. Various alternative to the accelerated algorithm were developed with different proofs [Nesterov, 2013*a*; Beck and Teboulle, 2009; Auslender and Teboulle, 2006; Chambolle et al., 1998] sometimes simpler [Tseng, 2008] but still difficult to interpret. Recent stream of papers use differential equations to model the acceleration behavior and offer another interpretation of Nesterov's algorithm [Su et al., 2014; Krichene et al., 2015; Wibisono et al., 2016; Wilson et al., 2016]. However, the differential equation is often quite complex, being reverse-engineered from Nesterov's method itself, thus losing the intuition. Moreover, integration methods for these differential equations are often ignored or are not derived from standard numerical integration schemes, since convergence proofs do not require the continuous-time interpretation.

In Scieur et al. [2017], we analyzed how optimization methods, in particular Nesterov's algorithm, can be interpreted as discretization methods of the simple gradient flow equation that reads

$$\dot{x}(t) = -\nabla f(x(t)), \quad x(0) = x_0. \tag{1.8}$$

For example gradient descent can be seen as an Euler's explicit scheme of integration for this differential equation. Proximal point algorithm [Rockafellar, 1976] translates as an Euler's implicit scheme. Finally extra-gradient algorithm, studied for example by [Nemirovski, 2004], that solves saddle point problems, is a predictor-corrector method [Press, 1992, §16.7] on the associated gradient flow equation.

In Scieur et al. [2017], we study specifically multi-step discretization methods for gradient flow equation derived from smooth and strongly convex functions. These schemes use several past points to build next iterate of the approximate solution of (1.8). Numerical analysis literature [Gautschi, 2011] provides conditions on the parameters to ensure consistency, i.e. convergence of the discrete approximation to the continuous solution of the differential equation on a finite time interval for infinitesimal step-sizes, which is essential for continuous time interpretation. Stability

for infinite time horizon can be derived for linear gradient flows (quadratic optimization) by some other conditions. Nesterov's algorithm and Polyak's heavy ball method can then be identified as consistent and stable multi-step methods of order 2 that allow bigger step-size of integration compared to simple Euler's scheme, which explains the acceleration phenomenon. Similar considerations applied for the non-strongly convex case by identifying accelerated method as a consistent integration scheme and identifying a bigger step-size. A full analysis of multi-step methods of order 2 is carried out for linear gradient flows (quadratic optimization). Future work is to certify stability of multi-step methods for smooth and strongly convex functions and explain why Heavy ball's method fails to solve some smooth strongly convex functions [Lessard et al., 2016].

Chapter 2

Sharpness, Restart and Acceleration

Chapter Abstract

The Łojasiewicz inequality shows that sharpness bounds on the minimum of convex optimization problems hold almost generically. Sharpness directly controls the performance of restart schemes, as observed by Nemirovskii and Nesterov [1985]. The constants quantifying error bounds are of course unobservable, but we show that optimal restart strategies are robust, and searching for the best scheme only increases the complexity by a logarithmic factor compared to the optimal bound. Overall then, restart schemes generically accelerate accelerated methods.

Introduction

We study convex optimization problems of the form

$$\text{minimize} \quad f(x) \tag{P}$$

where f is a convex function defined on $\mathbb{R}^n$. The complexity of these problems using first order methods is generically controlled by smoothness assumptions on f such as Lipschitz continuity of its gradient. Additional assumptions such as strong convexity or uniform convexity provide respectively linear [Nesterov, 2013*b*] and faster polynomial [Juditski and Nesterov, 2014] rates of convergence. However, these assumptions are often too restrictive to be applied. Here, we make a much weaker and generic assumption that describes the sharpness of the function around its minimizers by constants $\mu \geq 0$ and $r \geq 1$ such that

$$\frac{\mu}{r} d(x, X^*)^r \leq f(x) - f^*, \quad \text{for every } x \in K, \tag{Sharp}$$

where f^* is the minimum of f, $K \subset \mathbb{R}^n$ is a compact set, $d(x, X^*) = \min_{y \in X^*} \|x - y\|$ is the distance from x to the set $X^* \subset K$ of minimizers of f [1] for the Euclidean norm

1. We assume the problem feasible, i.e. $X^* \neq \emptyset$.

$\|\cdot\|$. This defines a *lower bound* on the function around its minimizers: for $r = 1$, f shows a kink around its minimizers and the larger is r the flatter is the function around its minimizers. We tackle this property by restart schemes of classical convex optimization algorithms.

Sharpness assumption (Sharp) is better known as a Hölderian error bound on the distance to the set of minimizers. Hoffman [Hoffman, 1952] first introduced error bounds to study system of linear inequalities. Natural extensions were then developed for convex optimization [Robinson, 1975; Mangasarian, 1985; Auslender and Crouzeix, 1988], notably through the concept of sharp minima [Polyak, 1979; Burke and Ferris, 1993; Burke and Deng, 2002]. But the most striking discovery was made by Łojasiewicz [Łojasiewicz, 1963, 1993] who proved inequality (Sharp) for real analytic and subanalytic functions. It has then been extended to non-smooth subanalytic convex functions by Bolte et al. [2007]. Overall, since (Sharp) essentially measures the sharpness of minimizers, it holds somewhat generically. On the other hand, this inequality is purely descriptive as we have no hope of ever observing either r or μ, and deriving adaptive schemes is crucial to ensure practical relevance.

Łojasiewicz inequalities either in the form of (Sharp) or as gradient dominated properties [Polyak, 1979] led to new simple convergence results [Karimi et al., 2016], in particular for alternating and splitting methods [Attouch et al., 2010; Frankel et al., 2015], even in the non-convex case [Bolte et al., 2014]. Here we focus on Hölderian error bounds as they offer simple explanation of accelerated rates of restart schemes.

Restart schemes were already studied for strongly or uniformly convex functions [Nemirovskii and Nesterov, 1985; Nesterov, 2013*a*; Juditski and Nesterov, 2014; Lin and Xiao, 2014]. In particular, Nemirovskii and Nesterov [1985] link a "strict minimum" condition akin to (Sharp) with faster convergence rates using restart schemes which form the basis of our results, but do not study the cost of adaptation and do not tackle the non-smooth case. In a similar spirit, weaker versions of this strict minimum condition were used more recently to study the performance of restart schemes in [Renegar, 2014; Freund and Lu, 2015; Roulet et al., 2015]. The fundamental question of a restart scheme is naturally to know when must an algorithm be stopped and relaunched. Several heuristics [O'Donoghue and Candes, 2015; Su et al., 2014; Giselsson and Boyd, 2014] studied adaptive restart schemes to speed up convergence of optimal methods. The robustness of restart schemes was then theoretically studied by Fercoq and Qu [2016] for quadratic error bounds, i.e. (Sharp) with $r = 2$, that LASSO problem satisfies for example. Fercoq and Qu [2017] extended recently their work to produce adaptive restarts with theoretical guarantees of optimal performance, still for quadratic error bounds. Previous references focus on smooth problems, but error bounds appear also for non-smooth ones, Gilpin et al. [2012] prove for example linear converge of restart schemes in bilinear matrix games where the minimum is sharp, i.e. (Sharp) with $r = 1$.

Our contribution here is to derive optimal scheduled restart schemes for general convex optimization problems for smooth, non-smooth or Hölder smooth functions satisfying the sharpness assumption. We then show that for smooth functions these schemes can be made adaptive with nearly optimal complexity (up to a squared log term) for a wide array of sharpness assumptions. We also analyze restart criterion

based on a sufficient decrease of the gap to the minimum value of the problem, when this latter is known in advance. In that case, restart schemes are shown to be optimal without requiring any additional information on the function.

The paper is organized as follows. In Section 2.1, we present the assumptions we make on the problem, i.e. smoothness and sharpness of the function, and discuss their link. In Section 2.2, scheduled restart strategies are presented for smooth unconstrained convex minimization problems satisfying the Łojasiewicz inequality, together with adaptive variants. In Section 2.3, these results are generalized to functions with Hölder continuous gradients. Adaptive restart schemes when the optimal value is known are introduced in Section 2.4. In Section 2.5, our results are extended to composite problems using Bregman divergences, and in particular prox-friendly constrained problems. Finally, numerical experiments are provided in Section 2.6.

Notations

For a real a, $\lceil a \rceil$ and $\lfloor a \rfloor$ denote respectively the smallest integer larger than or equal to a and the largest integer smaller than or equal to a.

2.1 Problem assumptions

In the following, we present the geometry of the problem with respect to the Euclidean norm, such that $d(x, X^*) = \min_{y \in X^*} \|x - y\|_2$ is the Euclidean distance from a point $x \in \mathbb{R}^n$ to the set of minimizers. In Section 2.5, we detail how our approach generalizes to other geometries handled by Bregman divergences.

2.1.1 Smoothness

Convex optimization problems (P) are generally divided in two classes : smooth problems, for which f has Lipschitz continuous gradients, and non-smooth problems for which f is not differentiable. Nesterov [2015] proposed to unify point of views by assuming generally that there exist constant $1 \leq s \leq 2$ and $L > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L \|x - y\|_2^{s-1}, \quad \text{for all } x, y \in \mathbb{R}^n \tag{Smooth}$$

where $\nabla f(x)$ is any sub-gradient of f at x if $s = 1$ (otherwise this implies differentiability of f). For $s = 2$, we retrieve the classical definition of smoothness [Nesterov, 2013*b*]. For $s = 1$ we get a classical assumption made in non-smooth convex optimization, i.e. that sub-gradients of the function are bounded. For $1 < s < 2$, this assumes gradient of f to be Hölder Lipschitz. In a first step, we will analyze restart schemes for smooth convex optimization problems, then generalize to general smoothness assumption (Smooth) using appropriate accelerated algorithms developed by Nesterov [2015].

2.1.2 Error bounds

In general, an error bound is an inequality of the form

$$d(x, X^*) \leq \omega(f(x) - f^*),$$

where ω is an increasing function at 0, called the residual function, and x may evolve either in the whole space or in a bounded set, see Bolte et al. [2015] for more details. We focus on Hölderian Error Bounds (Sharp) as they are the most common in practice, they are notably satisfied by a analytic and subanalytic functions but the proof (see e.g. Bierstone and Milman [1988]) is shown using topological arguments that are far from constructive, hence outside of some particular cases (e.g. strong convexity), we cannot assume that the constants in (Sharp) are known, even approximately.

Error bounds can generically be linked to Łojasiewicz inequality that upper bounds magnitude of the gradient by values of the function [Bolte et al., 2015]. Such property paved the way to many recent results in optimization [Attouch et al., 2010; Frankel et al., 2015; Bolte et al., 2014]. Here we will see that (Sharp) is sufficient to acceleration of convex optimization algorithms by their restart. Note finally that in most cases, error bounds are *local properties* hence the convergence results that follow will generally be local.

2.1.3 Sharpness and smoothness

Let f be a convex function on $\mathbb{R}^n$ satisfying (Smooth) with parameters (s, L). This property ensures that, $f(x) \leq f^* + \frac{L}{s}\|x - y\|_2^s$, for given $x \in \mathbb{R}^n$ and $y \in X^*$. Setting y to be the projection of x onto X^*, this yields the following *upper bound* on suboptimality

$$f(x) - f^* \leq \frac{L}{s} d(x, X^*)^s. \tag{2.1}$$

Now assume that f satisfies the error bound (Sharp) on a set K with parameters (r, μ). Combining (2.1) and (Sharp) this leads for every $x \in K$,

$$\frac{s\mu}{rL} \leq d(x, X^*)^{s-r}.$$

This means that necessarily $s \leq r$ by taking $x \to X^*$. Moreover if $s < r$, this last inequality can only be valid on a bounded set, i.e. either smoothness or error bound or both are valid only on a bounded set. In the following, we write

$$\kappa \triangleq L^{\frac{2}{s}}/\mu^{\frac{2}{r}} \qquad \text{and} \qquad \tau \triangleq 1 - \frac{s}{r} \tag{2.2}$$

respectively a generalized condition number for the function f and a condition number based on the ratio of powers in inequalities (Smooth) and (Sharp). If $r = s = 2$, κ matches the classical condition number of the function.

2.2 Scheduled restarts for smooth convex problems

In this section f is assumed to be smooth, i.e. satisfies (Smooth) with $s = 2$ and $L > 0$. Without further assumptions on f, an optimal algorithm to solve the smooth convex optimization problem (P) is Nesterov's accelerated gradient method [Nesterov, 1983]. Given an initial point x_0, this algorithm outputs, after t iterations, a point $x = \mathcal{A}(x_0, t)$ such that

$$f(x) - f^* \leq \frac{cL}{t^2} d(x_0, X^*)^2, \tag{2.3}$$

where $c > 0$ denotes a universal constant (whose value will be allowed to vary in what follows, with $c = 4$ here). We assume without loss of generality that $f(x) \leq f(x_0)$. More details about Nesterov's algorithm are given in Appendix A.2.

In what follows, we will also assume that f satisfies (Sharp) with parameters (r, μ) on a set $K \supseteq X^*$, which means

$$\frac{\mu}{r} d(x, X^*)^r \leq f(x) - f^*, \quad \text{for every } x \in K.$$

As mentioned before if $r > s = 2$, this property is necessarily local, i.e. K is bounded. We assume then that given a starting point $x_0 \in \mathbb{R}^n$, (Sharp) is satisfied on the sublevel set $\{x |\ f(x) \leq f(x_0)\}$. Remark that if this property is valid on an open set $K \supset X^*$, it will also be valid on any compact set $K' \supset K$ with the same exponent r but a potentially lower constant μ [Bierstone and Milman, 1988, Theorem 6.4]. The scheduled restart schemes presented assume (Sharp) on the whole sublevel set defined by the initial point and are not adaptive to the best local constant μ. On the other hand, restarts on criterion introduced in Section 2.4, assuming that f^* is known, adapt to the value of μ. We now describe a restart scheme exploiting this extra regularity assumption to improve the computational complexity of solving problem (P) using accelerated methods.

2.2.1 Scheduled restarts

Here we schedule the number of iterations t_k made by Nesterov's algorithm between restarts, with t_k the number of (inner) iterations at the k^{th} algorithm run (outer iteration). Our scheme is described below.

Algorithm 1 Scheduled restarts for smooth convex minimization **(RESTART)**

Inputs : $x_0 \in \mathbb{R}^n$ and a sequence t_k for $k = 1, \ldots, R$.
for $k = 1, \ldots, R$ **do**

$$x_k := \mathcal{A}(x_{k-1}, t_k) \tag{RESTART}$$

end for
Output : $\hat{x} := x_R$

The analysis of this scheme and the following ones relies on two steps : first choose schedules that ensure linear convergence in the iterates x_k at a given rate, then adjust this linear rate to minimize the complexity in terms of the total number of iterations.

We begin by a technical lemma which assumes linear convergence holds, and connects the growth of t_k, the precision reached and the total number of inner iterations N.

Lemma 2.2.1. *Let x_k be a sequence whose k^{th} iterate is generated from the previous one by an algorithm that runs t_k iterations and write $N = \sum_{k=1}^{R} t_k$ the total number of iterations to output a point x_R. Suppose setting $t_k = Ce^{\alpha k}$, $k = 1, \dots, R$ for some $C > 0$ and $\alpha \geq 0$ ensures that outer iterations satisfy*

$$f(x_k) - f^* \leq \nu e^{-\gamma k}, \tag{2.4}$$

for all $k \geq 0$ with $\nu \geq 0$ and $\gamma \geq 0$. Then precision at the output is given by,

$$f(x_R) - f^* \leq \nu \exp(-\gamma N/C), \quad \text{when } \alpha = 0,$$

and

$$f(x_R) - f^* \leq \frac{\nu}{(\alpha e^{-\alpha} C^{-1} N + 1)^{\frac{\gamma}{\alpha}}}, \quad \text{when } \alpha > 0.$$

Proof. When $\alpha = 0$, $N = RC$, and inserting this in (2.4) at the last point x_R yields the desired result. On the other hand, if $\alpha > 0$, then $N = \sum_{k=1}^{R} t_k = Ce^{\alpha} \frac{e^{\alpha R} - 1}{e^{\alpha} - 1}$, which gives $R = \log\left(\frac{e^{\alpha}-1}{e^{\alpha} C} N + 1\right) / \alpha$. Inserting this in (2.4) at the last point, this leads

$$f(x_R) - f^* \leq \nu \exp\left(-\frac{\gamma}{\alpha} \log\left(\frac{e^{\alpha} - 1}{e^{\alpha} C} N + 1\right)\right) \leq \frac{\nu}{(\alpha e^{-\alpha} C^{-1} N + 1)^{\frac{\gamma}{\alpha}}},$$

using that $e^x - 1 \geq x$. This yields the second part of the result. ∎

The last approximation in the case $\alpha > 0$ simplifies the analysis that follows without significantly affecting the bounds. We also show in Appendix 2.A that using $\tilde{t}_k = \lceil t_k \rceil$ does not significantly affect the bounds above. Remark that convergence bounds are generally linear or polynomial such that one can extract a subsequence that converges linearly. Therefore our approach does not restrict the analysis of our scheme. It simplifies it and can be used for other algorithms like the gradient descent as detailed in Section 2.2.3.

We now analyze restart schedules t_k that ensure linear convergence. Our choice of t_k will heavily depend on the ratio between r and s (with $s = 2$ for smooth functions here), incorporated in the parameter $\tau = 1 - s/r$ defined in (2.2). The following Proposition shows that if $\tau = 0$, a constant schedule is sufficient to ensure linear convergence. When $\tau > 0$, this requires a geometrically increasing number of iterations for each cycle.

Proposition 2.2.2. *Let f be a smooth convex function satisfying (Smooth) with parameters $(2, L)$ and (Sharp) with parameters (r, μ) on a set K. Assume that we*

are given $x_0 \in \mathbb{R}^n$ *such that* $\{x|\ f(x) \leq f(x_0)\} \subset K$. *Run* (RESTART) *from* x_0 *with iteration schedule* $t_k = C^*_{\kappa,\tau} e^{\tau k}$, *for* $k = 1, \ldots, R$, *where*

$$C^*_{\kappa,\tau} \triangleq e^{1-\tau}(c\kappa)^{\frac{1}{2}}(f(x_0) - f^*)^{-\frac{\tau}{2}}, \tag{2.5}$$

with κ *and* τ *defined in* (2.2) *and* $c = 4e^{2/e}$ *here. The precision reached at the last point* $\hat{x}$ *is given by,*

$$f(\hat{x}) - f^* \leq \exp\left(-2e^{-1}(c\kappa)^{-\frac{1}{2}}N\right)(f(x_0) - f^*) = O\left(\exp(-\kappa^{-\frac{1}{2}}N)\right), \quad \text{when } \tau = 0, \tag{2.6}$$

while,

$$f(\hat{x}) - f^* \leq \frac{f(x_0) - f^*}{\left(\tau e^{-1}(f(x_0) - f^*)^{\frac{\tau}{2}}(c\kappa)^{-\frac{1}{2}}N + 1\right)^{\frac{2}{\tau}}} = O\left(N^{-\frac{2}{\tau}}\right), \quad \text{when } \tau > 0, \tag{2.7}$$

where $N = \sum_{k=1}^{R} t_k$ *is the total number of iterations.*

Proof. Our strategy is to choose t_k such that the objective is linearly decreasing, i.e.

$$f(x_k) - f^* \leq e^{-\gamma k}(f(x_0) - f^*), \tag{2.8}$$

for some $\gamma \geq 0$ depending on the choice of t_k. This directly holds for $k = 0$ and any $\gamma \geq 0$. Combining (Sharp) with the complexity bound in (2.3) gives

$$f(x_k) - f^* \leq \frac{c\kappa}{t_k^2}(f(x_{k-1}) - f^*)^{\frac{2}{r}},$$

where $c = 4e^{2/e}$ using that $r^{2/r} \leq e^{2/e}$. Assuming recursively that (2.8) is satisfied at iteration $k-1$ for a given γ,

$$f(x_k) - f^* \leq \frac{c\kappa e^{-\gamma\frac{2}{r}(k-1)}}{t_k^2}(f(x_0) - f^*)^{\frac{2}{r}},$$

and to ensure (2.8) at iteration k, this imposes

$$\frac{c\kappa e^{-\gamma\frac{2}{r}(k-1)}}{t_k^2}(f(x_0) - f^*)^{\frac{2}{r}} \leq e^{-\gamma k}(f(x_0) - f^*).$$

Rearranging terms in this last inequality, using τ defined in (2.2),

$$t_k \geq e^{\frac{\gamma(1-\tau)}{2}}(c\kappa)^{\frac{1}{2}}(f(x_0) - f^*)^{-\frac{\tau}{2}}e^{\frac{\tau\gamma}{2}k}. \tag{2.9}$$

For a given $\gamma \geq 0$, we can set $t_k = Ce^{\alpha k}$ where

$$C = e^{\frac{\gamma(1-\tau)}{2}}(c\kappa)^{\frac{1}{2}}(f(x_0) - f^*)^{-\frac{\tau}{2}} \qquad \text{and} \qquad \alpha = \tau\gamma/2, \tag{2.10}$$

and Lemma 2.2.1 then yields,

$$f(\hat{x}) - f^* \leq \exp\left(-\gamma e^{-\frac{\gamma}{2}}(c\kappa)^{-\frac{1}{2}}N\right)(f(x_0) - f^*),$$

when $\tau = 0$, while

$$f(\hat{x}) - f^* \leq \frac{(f(x_0) - f^*)}{\left(\frac{\tau}{2}\gamma e^{-\frac{\gamma}{2}}(c\kappa)^{-\frac{1}{2}}(f(x_0) - f^*)^{\frac{\tau}{2}}N + 1\right)^{\frac{2}{\tau}}},$$

when $\tau > 0$. These bounds are minimal for $\gamma = 2$, which yields the desired result. ∎

When $\tau = 0$, bound (2.6) matches the classical complexity bound for smooth strongly convex functions [Nesterov, 2013*b*]. When $\tau > 0$ on the other hand, bound (2.7) highlights a *much faster convergence rate than accelerated gradient methods.* The sharper the function (i.e. the smaller r), the faster the convergence. This matches the lower bounds for optimizing smooth and sharp functions functions [Nemirovskii and Nesterov, 1985, Page 6] up to constant factors. Also, setting $t_k = C^*_{\kappa,\tau}e^{\tau k}$ yields continuous bounds on precision, i.e. when $\tau \to 0$, bound (2.7) converges to bound (2.6), which also shows that for τ near zero, constant restart schemes are almost optimal.

2.2.2 Adaptive scheduled restart

The previous restart schedules depend on the parameters (r, μ) in (Sharp). In general of course, these values are neither observed nor known a priori. Making our restart scheme adaptive is thus crucial to its practical performance. Fortunately, we show below that a simple logarithmic grid search strategy on these parameters is enough to guarantee nearly optimal performance.

In that purpose we need first the following Corollary of Proposition 2.2.2.

Corollary 2.2.3. *Let f be a smooth convex function satisfying (Smooth) with parameters $(2, L)$ and (Sharp) with parameters (r, μ) on a set K. Assume that we are given $x_0 \in \mathbb{R}^n$ such that $\{x : f(x) \leq f(x_0)\} \subset K$. Run (RESTART) from x_0 with general schedules of the form*

$$\begin{cases} t_k = C & \text{if } \tau = 0, \\ t_k = Ce^{\alpha k} & \text{if } \tau > 0, \end{cases}$$

we have the following complexity bounds, if $\tau = 0$ and $C \geq C^*_{\kappa,0}$,

$$f(\hat{x}) - f^* \leq \left(\frac{c\kappa}{C^2}\right)^{\frac{N}{C}}(f(x_0) - f^*), \tag{2.11}$$

while, if $\tau > 0$ and $C \geq C(\alpha)$,

$$f(\hat{x}) - f^* \leq \frac{f(x_0) - f^*}{(\alpha e^{-\alpha}C^{-1}N + 1)^{\frac{2}{\tau}}}, \tag{2.12}$$

where

$$C(\alpha) \triangleq e^{\frac{\alpha(1-\tau)}{\tau}}(c\kappa)^{\frac{1}{2}}(f(x_0)-f^*)^{-\frac{\tau}{2}}, \tag{2.13}$$

and $N = \sum_{k=1}^{R} t_k$ *is the total number of iterations.*

Proof. Given general schedules of the form

$$\begin{cases} t_k = C & \text{if } \tau = 0, \\ t_k = Ce^{\alpha k} & \text{if } \tau > 0, \end{cases}$$

the best value of γ satisfying condition (2.9) for any $k \geq 0$ in Proposition 2.2.2 are given by

$$\begin{cases} \gamma = \log\left(\frac{C^2}{c\kappa}\right) & \text{if } \tau = 0 \text{ and } C \geq C^*_{\kappa,0}, \\ \gamma = \frac{2\alpha}{\tau} & \text{if } \tau > 0 \text{ and } C \geq C(\alpha). \end{cases}$$

As in Proposition 2.2.2, plugging these values into the bounds of Lemma 2.2.1 yields the desired result. ■

This Corollary shows that scheduled restarts are theoretically efficient only if the algorithm itself makes a sufficient number of iterations. With this Corollary, an analysis of a grid search of the schedules can be made.

We run several schemes with a fixed number of inner iterations N to perform a log-scale grid search on τ and κ. These schemes are defined as follows.

$$\begin{cases} \mathcal{S}_{i,0} : \text{(RESTART) scheme with } t_k = C_i, \\ \mathcal{S}_{i,j} : \text{(RESTART) scheme with } t_k = C_i e^{\tau_j k}, \end{cases} \tag{2.14}$$

where $C_i = 2^i$ and $\tau_j = 2^{-j}$. These schemes are stopped when the total number of inner algorithm iterations has exceed N, i.e. at the smallest R such that $\sum_{k=1}^{R} t_k \geq N$. The size of the grid search in C_i is naturally bounded as the algorithm cannot be restarted after more than N total inner iterations, so $i \in [1, \dots, \lfloor \log_2 N \rfloor]$. We will also show that when τ is smaller than $1/N$, a constant schedule performs as well as the optimal geometrically increasing schedule, which crucially means that we can also choose $j \in [1, \dots, \lceil \log_2 N \rceil]$ and limits the cost of grid search.

Proposition 2.2.4. *Let* f *be a smooth convex function satisfying* (Smooth) *with parameters* $(2, L)$ *and* (Sharp) *with parameters* (r, μ) *on a set* K*. Assume that we are given* $x_0 \in \mathbb{R}^n$ *such that* $\{x |\ f(x) \leq f(x_0)\} \subset K$ *and denote* N *a given number of iterations. Run schemes* $\mathcal{S}_{i,j}$ *defined in* (2.14) *to solve* (P) *for* $i \in [1, \dots, \lfloor \log_2 N \rfloor]$ *and* $j \in [0, \dots, \lceil \log_2 N \rceil]$*, stopping each time after* N *total inner algorithm iterations i.e. for* R *such that* $\sum_{k=1}^{R} t_k \geq N$*.*

Assume N *is large enough, so* $N \geq 2C^*_{\kappa,\tau}$*, and if* $\frac{1}{N} > \tau > 0$*,* $C^*_{\kappa,\tau} > 1$*.*

If $\tau = 0$*, there exists* $i \in [1, \dots, \lfloor \log_2 N \rfloor]$ *such that scheme* $\mathcal{S}_{i,0}$ *achieves a precision given by*

$$f(\hat{x}) - f^* \leq \exp\left(-e^{-1}(c\kappa)^{-\frac{1}{2}}N\right)(f(x_0) - f^*).$$

If $\tau > 0$*, there exist* $i \in [1, \dots, \lfloor \log_2 N \rfloor]$ *and* $j \in [1, \dots, \lceil \log_2 N \rceil]$ *such that*

scheme $\mathcal{S}_{i,j}$ achieves a precision given by

$$f(\hat{x}) - f^* \leq \frac{f(x_0) - f^*}{\left(\tau e^{-1}(c\kappa)^{-\frac{1}{2}}(f(x_0) - f^*)^{\frac{\tau}{2}}(N-1)/4 + 1\right)^{\frac{2}{\tau}}}.$$

Overall, running the logarithmic grid search has a complexity $(\log_2 N)^2$ times higher than running N iterations using the optimal (oracle) scheme.

Proof. Denote $N' = \sum_{k=1}^{R} t_k \geq N$ the number of iterations of a scheme $\mathcal{S}_{i,j}$. We necessarily have $N' \leq 2N$ for our choice of C_i and τ_j. Hence the cost of running all methods is of the order $(\log_2 N)^2$.

If $\tau = 0$ and $N \geq 2C^*_{\kappa,0}$, then $i = \lceil \log_2 C^*_{\kappa,0} \rceil \leq \lfloor \log_2 N \rfloor$. Therefore $\mathcal{S}_{i,0}$ has been run and bound (2.11) to shows then that the last iterate $\hat{x}$ satisfies

$$f(\hat{x}) - f^* \leq \left(\frac{c\kappa}{C_i^2}\right)^{\frac{N}{C_i}} (f(x_0) - f^*).$$

Using that $C^*_{\kappa,0} \leq C_i \leq 2C^*_{\kappa,0}$,

$$\begin{aligned} f(\hat{x}) - f^* &\leq \left(\frac{c\kappa}{(C^*_{\kappa,0})^2}\right)^{\frac{N}{2C^*_{\kappa,0}}} (f(x_0) - f^*) \\ &\leq \exp\left(-e^{-1}(c\kappa)^{-\frac{1}{2}} N\right) (f(x_0) - f^*). \end{aligned}$$

If $\tau \geq \frac{1}{N}$ and $N \geq 2C^*_{\kappa,\tau}$, then $j = \lceil -\log_2 \tau \rceil \leq \lceil \log_2 N \rceil$ and $i = \lceil \log_2 C^*_{\kappa,\tau} \rceil \leq \lfloor \log_2 N \rfloor$. Therefore scheme $\mathcal{S}_{i,j}$ has been run. As $C_i \geq C^*_{\kappa,\tau} \geq C(\tau_j)$, where $C(\tau_j)$ is defined in (2.13), bound (2.12) shows then that the last iterate $\hat{x}$ of scheme $\mathcal{S}_{i,j}$ satisfies

$$f(\hat{x}) - f^* \leq \frac{f(x_0) - f^*}{\left(\tau_j e^{-\tau_j} C_i^{-1} N + 1\right)^{\frac{2}{\tau}}}.$$

Finally, by definition of i and j, $2\tau_j \geq \tau$ and $C_i \leq 2C^*_{\kappa,\tau}$, so

$$\begin{aligned} f(\hat{x}) - f^* &\leq \frac{f(x_0) - f^*}{\left(\tau e^{-\tau_j}(C^*_{\kappa,\tau})^{-1} N/4 + 1\right)^{\frac{2}{\tau}}} \\ &= \frac{f(x_0) - f^*}{\left(\tau e^{-1}(c\kappa)^{-\frac{1}{2}}(f(x_0) - f^*)^{\frac{\tau}{2}} N/4 + 1\right)^{\frac{2}{\tau}}}, \end{aligned}$$

where we concluded by expanding $C^*_{\kappa,\tau} = e^{1-\tau}(c\kappa)^{\frac{1}{2}}(f(x_0) - f^*)^{-\frac{\tau}{2}}$ and using that $\tau \geq \tau_j$.

If $\frac{1}{N} > \tau > 0$ and $N > 2C^*_{\kappa,\tau}$, then $i = \lceil \log_2 C^*_{\kappa,\tau} \rceil \leq \lfloor \log_2 N \rfloor$, so scheme $\mathcal{S}_{i,0}$ has

been run. Its iterates x_k satisfy, with $1-\tau = 2/r$,

$$\begin{aligned} f(x_k) - f^* &\leq \frac{c\kappa}{C_i^2}(f(x_{k-1}) - f^*)^{\frac{2}{r}} \\ &\leq \left(\frac{c\kappa}{C_i^2}\right)^{\left(1-(1-\tau)^k\right)/\tau} (f(x_0) - f^*)^{(1-\tau)^k} \\ &\leq \left(\frac{c\kappa(f(x_0) - f^*)^{-\tau}}{C_i^2}\right)^{\left(1-(1-\tau)^k\right)/\tau} (f(x_0) - f^*). \end{aligned}$$

Now $C_i \geq C^*_{\kappa,\tau} = e^{1-\tau}(c\kappa)^{\frac{1}{2}}(f(x_0) - f^*)^{-\frac{\tau}{2}}$ and $C_i R \geq N$, therefore last iterate $\hat{x}$ satisfies

$$f(\hat{x}) - f^* \leq \exp\left(-2(1-\tau)\frac{1-(1-\tau)^{N/C_i}}{\tau}\right)(f(x_0) - f^*).$$

As $N \geq C_i$, since

$$h(\tau) = \frac{(1-\tau)\left(1-(1-\tau)^{\frac{N}{C_i}}\right)}{1-(1-\tau)}$$

is decreasing with τ and $\frac{1}{N} > \tau > 0$, we have

$$\begin{aligned} f(\hat{x}) - f^* &\leq \exp\left(-2(N-1)\left(1-\left(1-\frac{1}{N}\right)^{N/C_i}\right)\right)(f(x_0) - f^*) \\ &\leq \exp\left(-2(N-1)\left(1-\exp\left(-\frac{1}{C_i}\right)\right)\right)(f(x_0) - f^*) \\ &\leq \exp\left(-2\frac{N-1}{C_i}\left(1-\frac{1}{2C_i}\right)\right)(f(x_0) - f^*). \end{aligned}$$

having used the facts that $(1+ax)^{\frac{b}{x}} \leq \exp(ab)$ if $ax \geq -1$, $\frac{b}{x} \geq 0$ and $1 - x + \frac{x^2}{2} \geq \exp(-x)$ when $x \geq 0$. By assumption $C^*_{\kappa,\tau} \geq 1$, so $C_i \geq 1$ and finally

$$\begin{aligned} f(\hat{x}) - f^* &\leq \exp\left(-\frac{N-1}{C_i}\right)(f(x_0) - f^*) \\ &\leq \exp\left(-\frac{N-1}{2C^*_{\kappa,\tau}}\right)(f(x_0) - f^*) \\ &\leq \frac{f(x_0) - f^*}{\left(\tau(C^*_{\kappa,\tau})^{-1}(N-1)/4 + 1\right)^{\frac{2}{\tau}}} \\ &\leq \frac{f(x_0) - f^*}{\left(\tau(f(x_0) - f^*)^{\frac{\tau}{2}}e^{-1}(c\kappa)^{-\frac{1}{2}}(N-1)/4 + 1\right)^{\frac{2}{\tau}}}. \end{aligned}$$

using the fact that $e^\tau \geq 1$. ■

As shown in Corollary 2.2.3, theoretical efficiency of scheduled restarts is ensured only for a sufficient number of iterations of the algorithm itself. Therefore N needs to be large enough to ensure the efficiency of the adaptive method. If $\tau = 0$, then naturally $C^*_{\kappa,0} \geq 1$, therefore if $\frac{1}{N} > \tau > 0$ and N is large, assuming $C^*_{\kappa,\tau} \approx C^*_{\kappa,0}$, it results $C^*_{\kappa,\tau} \geq 1$. This adaptive bound is similar to the one of Nesterov [2013*a*] to optimize smooth strongly convex functions in the sense that we lose approximately a log factor of the condition number of the function. However our assumptions are weaker and we are able to tackle all regimes of the sharpness property, i.e. any exponent $r \in [2, +\infty]$, not just the strongly convex case. Finally the step size chosen for the grid search was set to 2. Proof can adapted for a generic step size h, the size of the grid may be reduced but corresponding bounds will suffer an h^2 approximation loss compared to the best schedule.

We end this section by analyzing the behavior of gradient descent in light of the sharpness assumption.

2.2.3 Comparison to gradient descent

Given only the smoothness hypothesis, the gradient descent algorithm, recalled in Appendix A.3, starts from a point x_0 and outputs iterates $x_t = \mathcal{G}(x_0, t)$ such that

$$f(x_t) - f^* \leq \frac{L}{t} d(x_0, X^*)^2,$$

While accelerated methods use the last two iterates to compute the next one, simple gradient descent algorithms use only the last iterate, so the algorithm can be seen as (implicitly) restarting at each iteration. Its convergence can therefore be written for $k \geq 1$,

$$f(x_{k+t}) - f^* \leq \frac{L}{t} d(x_k, X^*)^2. \tag{2.15}$$

and we analyze it in light of the restart interpretation using the error bound in the following proposition.

Proposition 2.2.5. *Let f be a smooth convex function satisfying (Smooth) with parameters $(2, L)$ and (Sharp) with parameters (r, μ) on a set K. Assume that we are given $x_0 \in \mathbb{R}^n$ such that $\{x |\, f(x) \leq f(x_0)\} \subset K$. Denote $x_t = \mathcal{G}(x_0, t)$ the iterate sequence generated by the gradient descent algorithm started at x_0 to solve (P). Define*

$$t_k = e^{1-\tau} c\kappa (f(x_0) - f^*)^\tau e^{\tau k},$$

with κ and τ defined in (2.2) and $c = e^{2/e}$ here. The precision reached after $N = \sum_{k=1}^n t_k$ iterations is given by,

$$f(x_N) - f^* \leq \exp\left(-e^{-1}(c\kappa)^{-1} N\right) (f(x_0) - f^*) = O\left(\exp(-\kappa^{-1} N)\right), \quad \text{when } \tau = 0,$$

while,

$$f(x_N) - f^* \leq \frac{f(x_0) - f^*}{(\tau e^{-1}(c\kappa)^{-1}(f(x_0) - f^*)^\tau N + 1)^{\frac{1}{\tau}}} = O\left(N^{-\frac{1}{\tau}}\right), \quad \textit{when } \tau > 0.$$

Proof. For a given $\gamma \geq 0$, we construct a subsequence $x_{\phi(k)}$ of x_t such that

$$f(x_{\phi(k)}) - f^* \leq e^{-\gamma k}(f(x_0) - f^*). \tag{2.16}$$

Define $x_{\phi(0)} = x_0$. Assume that (2.16) is true at iteration $k-1$, then combining complexity bound (2.15) and (Sharp), for any $t \geq 1$,

$$\begin{aligned} f(x_{\phi(k-1)+t}) - f^* &\leq \frac{c\kappa}{t}(f(x_{\phi(k-1)}) - f^*)^{\frac{2}{r}} \\ &\leq \frac{c\kappa}{t} e^{-\gamma\frac{2}{r}(k-1)}(f(x_0) - f^*)^{\frac{2}{r}}. \end{aligned}$$

where $c = e^{2/e}$, using that $r^{2/r} \leq e^{2/e}$. Taking $t_k = e^{\gamma(1-\tau)}c\kappa(f(x_0) - f^*)^{-\tau}e^{\gamma\tau k}$ and $\phi(k) = \phi(k-1) + t_k$, (2.16) holds at iteration k. Using Lemma 2.2.1, we obtain at iteration $N = \phi(n) = \sum_{k=1}^n t_k$,

$$f(x_N) - f^* \leq \exp\left(-\gamma e^{-\gamma}(c\kappa)^{-1}N\right)(f(x_0) - f^*), \quad \text{if } \tau = 0,$$

and

$$f(x_N) - f^* \leq \frac{f(x_0) - f^*}{(\tau\gamma e^{-\gamma}(c\kappa)^{-1}(f(x_0) - f^*)^\tau N + 1)^{\frac{1}{\tau}}}, \quad \text{if } \tau > 0.$$

These bounds are minimal for $\gamma = 1$ and the results follow. ∎

We observe that restarting accelerated gradient methods reduces complexity from $O(1/\varepsilon^\tau)$ to $O(1/\varepsilon^{\tau/2})$ compared to simple gradient descent. More general results on the convergence of (sub)gradient descent algorithms under a Łojasiewicz inequality assumption were developed by Bolte et al. [2015]. We extend now this restart scheme to solve general convex optimization problem under an Hölderian error bound assumption.

2.3 Universal scheduled restarts for convex functions

In this section we use the framework introduced by Nesterov [2015] to describe smoothness of a function f on a set $J \subset \mathbb{R}^n$. We recall that it assumes that there exist $s \in [1, 2]$ and $L > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|^{s-1}, \quad \text{for every } x, y \in J,$$

If the function is non smooth, it satisfies (Smooth) with $s = 1$ and L taken as the maximum norm of subgradients on J. Without further assumptions on f, an optimal algorithm to solve the convex optimization problem (P) is the universal fast gradi-

ent method [Nesterov, 2015]. Given a target accuracy ε, the universal fast gradient method starts at a point x_0 and outputs after t iterations a point $x \triangleq \mathcal{U}(x_0, \varepsilon, t)$, such that

$$f(x) - f^* \leq \frac{\varepsilon}{2} + \frac{cL^{\frac{2}{s}} d(x_0, X^*)^2}{\varepsilon^{\frac{2}{s}} t^{\frac{2\rho}{s}}} \frac{\varepsilon}{2}, \tag{2.17}$$

where c is a constant ($c = 8$) and

$$\rho \triangleq \frac{3s}{2} - 1 \tag{2.18}$$

is the optimal rate of convergence for s-smooth functions. More details about the universal fast gradient method are given in Appendix A.1.

We will again assume that f satisfies (Sharp) with parameters (r, μ) on a set $K \supseteq X^*$, i.e.

$$\frac{\mu}{r} d(x, X^*)^r \leq f(x) - f^*, \quad \text{for every } x \in K.$$

As mentioned in Section 2.1, if $r > s$, smoothness or sharpness are local properties, i.e. either J or K or both are bounded, our analysis is therefore local. In the following we assume for simplicity, given an initial point x_0, that smoothness and sharpness are satisfied simultaneously on the sublevel set $\{x | f(x) \leq f(x_0)\}$. The key difference with the smooth case described in the previous section is that here we schedule *both* the target accuracy ε_k used by the algorithm *and* the number of iterations t_k made at the k^{th} run of the algorithm. Our scheme is described in Algorithm Universal RESTART.

Algorithm 2 General scheduled restarts for convex minimization **(Universal RESTART)**

Inputs : $x_0 \in \mathbb{R}^n$, $\varepsilon_0 \geq f(x_0) - f^*$, $\gamma \geq 0$ and a sequence t_k for $k = 1, \ldots, R$.
for $k = 1, \ldots, R$ **do**

$$\varepsilon_k := e^{-\gamma} \varepsilon_{k-1}, \qquad x_k := \mathcal{U}(x_{k-1}, \varepsilon_k, t_k) \qquad \text{(Universal RESTART)}$$

end for
Output : $\hat{x} := x_R$

Our strategy is to choose a sequence t_k that ensures

$$f(x_k) - f^* \leq \varepsilon_k,$$

for the geometrically decreasing sequence ε_k. The overall complexity of our method will then depend on the growth of t_k as described in Lemma 2.2.1.

Proposition 2.3.1. *Let f be a convex function satisfying (Smooth) with parameter (s, L) on a set J and (Sharp) with parameters (r, μ) on a set K. Given $x_0 \in \mathbb{R}^n$ assume that $\{x | f(x) \leq f(x_0)\} \subset J \cap K$. Run (Universal RESTART) scheme from x_0*

for a given $\varepsilon_0 \geq f(x_0) - f^*$ *with*

$$\gamma = \rho, \qquad t_k = C^*_{\kappa,\tau,\rho} e^{\tau k}, \quad \text{where} \quad C^*_{\kappa,\tau,\rho} \triangleq e^{1-\tau}(c\kappa)^{\frac{s}{2\rho}} \varepsilon_0^{-\frac{\tau}{\rho}}$$

where ρ *is defined in* (2.18), κ *and* τ *are defined in* (2.2) *and* $c = 8e^{2/e}$ *here. The precision reached at the last point* $\hat{x}$ *is given by,*

$$f(\hat{x}) - f^* \;\leq\; \exp\left(-\rho e^{-1}(c\kappa)^{-\frac{s}{2\rho}} N\right) \varepsilon_0 \;=\; O\left(\exp(-\kappa^{-\frac{s}{2\rho}} N)\right), \quad \text{when } \tau = 0,$$

while,

$$f(\hat{x}) - f^* \;\leq\; \frac{\varepsilon_0}{\left(\tau e^{-1}(c\kappa)^{-\frac{s}{2\rho}} \varepsilon_0^{\frac{\tau}{\rho}} N + 1\right)^{-\frac{\rho}{\tau}}} \;=\; O\left(\kappa^{\frac{s}{2\tau}} N^{-\frac{\rho}{\tau}}\right), \quad \text{when } \tau > 0,$$

where $N = \sum_{k=1}^{R} t_k$ *is total number of iterations.*

Proof. Our goal is to ensure that the target accuracy is reached at each restart, i.e.

$$f(x_k) - f^* \leq \varepsilon_k. \tag{2.19}$$

By assumption, (2.19) holds for $k = 0$. Assume that (2.19) is true at iteration $k-1$, combining (Sharp) with the complexity bound in (2.17), then

$$\begin{aligned} f(x_k) - f^* &\leq \frac{\varepsilon_k}{2} + \frac{c\kappa (f(x_{k-1}) - f^*)^{\frac{2}{r}}}{\varepsilon_k^{\frac{2}{s}} t_k^{\frac{2\rho}{s}}} \frac{\varepsilon_k}{2} \\ &\leq \frac{\varepsilon_k}{2} + \frac{c\kappa}{t_k^{\frac{2\rho}{s}}} \frac{\varepsilon_{k-1}^{\frac{2}{r}}}{\varepsilon_k^{\frac{2}{s}}} \frac{\varepsilon_k}{2}, \end{aligned}$$

where $c = 8e^{2/e}$ using that $r^{2/r} \leq e^{2/e}$. By definition $\varepsilon_k = e^{-\gamma k}\varepsilon_0$, so to ensure (2.19) at iteration k this imposes

$$\frac{c\kappa e^{\gamma\frac{2}{r}} e^{-\gamma\left(\frac{2}{r}-\frac{2}{s}\right)k}}{t_k^{\frac{2\rho}{s}}} \varepsilon_0^{\frac{2}{r}-\frac{2}{s}} \leq 1.$$

Rearranging terms in last inequality, using τ defined in (2.2),

$$t_k \geq e^{\gamma\frac{1-\tau}{\rho}} (c\kappa)^{\frac{s}{2\rho}} \varepsilon_0^{-\frac{\tau}{\rho}} e^{\frac{\gamma\tau}{\rho}k}.$$

Choosing $t_k = Ce^{\alpha k}$, where

$$C = e^{\gamma\frac{1-\tau}{\rho}} (c\kappa)^{\frac{s}{2\rho}} \varepsilon_0^{-\frac{\tau}{\rho}} \qquad \text{and} \qquad \alpha = \frac{\gamma\tau}{\rho},$$

and using Lemma 2.2.1 then yields,

$$f(\hat{x}) - f^* \leq \exp(-\gamma e^{-\frac{\gamma}{\rho}}(c\kappa)^{-\frac{s}{2\rho}}N)\varepsilon_0, \tag{2.20}$$

when $\tau = 0$, while,

$$f(\hat{x}) - f^* \leq \frac{\varepsilon_0}{\left(\frac{\gamma\tau}{\rho}e^{-\frac{\gamma}{\rho}}(c\kappa)^{-\frac{s}{2\rho}}\varepsilon_0^{\frac{\tau}{\rho}}N + 1\right)^{\frac{\rho}{\tau}}}. \tag{2.21}$$

when $\tau > 0$. These bounds are minimal for $\gamma = \rho$ and the results follow. ∎

This bound matches the lower bounds for optimizing smooth and sharp functions [Nemirovskii and Nesterov, 1985, Page 6] up to constant factors. However, the rate of convergence of this method is controlled by the ratio between τ and ρ. If these are unknown, a log-scale grid search will not be able to reach the optimal rate even if ρ is known since we will miss the optimal rate, controlled by the ratio ρ/τ, by a constant factor. If both are known, in the case of non-smooth strongly convex functions for example, a grid-search on C recovers nearly the optimal bound. Now we will see that if f^* is known, restart produces adaptive optimal rates.

2.4 Restart with termination criterion

Here, we assume that we know the optimum f^* of (P), or have an exact termination criterion. This is the case for example in zero-sum matrix games problems. We assume again that f satisfies (Smooth) with parameters (s, L) on a set J and (Sharp) with parameters (r, μ) on a set K. Given an initial point x_0 we assume that smoothness and error bound are satisfied simultaneously on the sublevel set $\{x|\ f(x) \leq f(x_0)\}$. We use again the universal gradient method $\mathcal{U}$. Here however, we can stop the algorithm when it reaches the target accuracy as we know the optimum f^*, i.e. we stop after t_ε inner iterations such that $x = \mathcal{U}(x_0, \varepsilon, t_\varepsilon)$ satisfies $f(x) - f^* \leq \varepsilon$, and write the output of this method

$$x \triangleq \mathcal{C}(x_0, \varepsilon).$$

Here we simply restart this method and decrease the target accuracy by a constant factor after each restart. Our scheme is described in Algorithm ε-RESTART.

Algorithm 3 Restart on criterion **(ε-RESTART)**

Inputs : $x_0 \in \mathbb{R}^n, f^*, \gamma \geq 0, \varepsilon_0 = f(x_0) - f^*$
for $k = 1, \ldots, R$ **do**

$$\varepsilon_k := e^{-\gamma}\varepsilon_{k-1}, \qquad x_k := \mathcal{C}(x_{k-1}, \varepsilon_k) \tag{ε-RESTART}$$

end for
Output : $\hat{x} := x_R$

The following result describes the convergence of this method. It relies on the idea that it cannot do more iterations than the best scheduled restart to achieve the target accuracy at each iteration.

Proposition 2.4.1. *Let f be a convex function satisfying (Smooth) with parameter (s, L) on a set Q and (Sharp) with parameters (r, μ) on a set K. Given $x_0 \in \mathbb{R}^n$ assume that $\{x,\ f(x) \leq f(x_0)\} \subset Q \cap K$. Run ($\varepsilon$-RESTART) scheme from x_0 with parameter $\gamma = \rho$. The precision reached at the point x_R is given by,*

$$f(\hat{x}) - f^* \ \leq\ \exp\left(-\rho e^{-1}(c\kappa)^{-\frac{s}{2\rho}} N\right)(f(x_0) - f^*) \ =\ O\left(\exp(-\kappa^{-\frac{s}{2\rho}} N)\right), \quad \textit{when } \tau = 0,$$

while,

$$f(\hat{x}) - f^* \ \leq\ \frac{f(x_0) - f^*}{\left(\tau e^{-1}(c\kappa)^{-\frac{s}{2\rho}}(f(x_0) - f^*)^{\frac{\tau}{\rho}} N + 1\right)^{\frac{\rho}{\tau}}} \ =\ O\left(\kappa^{\frac{s}{2\tau}} N^{-\frac{\rho}{\tau}}\right), \quad \textit{when } \tau > 0,$$

where N is the total number of iterations.

Proof. Given $\gamma \geq 0$, linear convergence of our scheme is ensured by our choice of target accuracies ε_k. It remains to compute the number of iterations t_{ε_k} needed by the algorithm before the k^{th} restart. Following proof of Proposition 2.3.1, for $k \geq 1$ we know that target accuracy is necessarily reached after

$$\bar{t}_k = e^{\gamma \frac{1-\tau}{\rho}} (c\kappa)^{\frac{s}{2\rho}} \varepsilon_0^{-\frac{\tau}{\rho}} e^{\frac{\gamma\tau}{\rho} k}$$

iterations, such that $t_{\varepsilon_k} \leq \bar{t}_k$. So ($\varepsilon$-RESTART) scheme achieves linear convergence while needing less inner iterates than the scheduled restart presented in Proposition 2.3.1, its convergence is therefore at least as good. For a given γ bounds (2.20) and (2.21) follow with $\varepsilon_0 = f(x_0) - f^*$ and taking $\gamma = \rho$ is optimal. ∎

Therefore if f^* is known, this method is adaptive to μ and r, contrary to the general case in Proposition 2.3.1. It can even adapt to the local values of L or μ as we use a criterion instead of a preset schedule. Here, stopping using $f(x_k) - f^*$ implicitly yields optimal choices of C and τ. A closer look at the proof shows that the dependency in γ of this restart scheme is a factor $h(\gamma) = \gamma e^{-\gamma/\rho}$ of the number of iterations. Taking $\gamma = 1$, leads then to a suboptimal constant factor of at most $h(\rho)/h(1) \leq e/2 \approx 1.3$ for $\rho \in [1/2, 2]$, so running this scheme with $\gamma = 1$ makes it parameter-free while getting nearly optimal bounds.

2.5 Composite problems & Bregman divergences

The restart schemes detailed so far focused on unconstrained problems in an Euclidean setting. Here, we extend them to more general convex optimization problems of the form

$$\text{minimize } f(x) \triangleq \phi(x) + g(x), \qquad \text{(Composite)}$$

where ϕ is a convex function whose smoothness is described by parameters (L, s), such that

$$\|\nabla\phi(x) - \nabla\phi(y)\|_* \leq L\|x - y\|^{s-1} \quad \text{for every } x, y \in J, \qquad \text{(Generic Smooth)}$$

for a given norm $\|\cdot\|$ where $\|\cdot\|_*$ is its dual norm, and g is a simple convex function (the meaning of simple will be clarified later).

To exploit the smoothness of ϕ with respect to a generic norm, we assume that we have access to a prox function h with $\mathbf{dom}(f) \subset \mathbf{dom}(h)$, strongly convex with respect to the norm $\|\cdot\|$ with convexity parameter equal to one, which means

$$h(y) \geq h(x) + \nabla h(x)^T(y - x) + \frac{1}{2}\|x - y\|^2, \quad \text{for any } x, y \in \mathbf{dom}(h).$$

We define the Bregman divergence associated to h as

$$D_h(y, x) = h(y) - h(x) - \nabla h(x)^T(y - x), \quad \text{for } x, y \in \mathbf{dom}(h),$$

so that $D_h(y, x) \geq \frac{1}{2}\|x - y\|^2$. For $h(x) = \frac{1}{2}\|x\|_2^2$, we get $D_h(y, x) = \frac{1}{2}\|x - y\|_2^2$ and recover the Euclidean setting. Given the problem geometry, appropriate choices of prox functions and associated Bregman divergences can lead to significant performance gains in high dimensional settings.

We now formally state the assumption that g is simple. Given $x, y \in \mathbf{dom}(f)$ and $\lambda \geq 0$ we assume that

$$\min_z \left\{y^T z + g(z) + \lambda D_h(z, x)\right\}$$

can be solved either in a closed form or by some fast computational procedure. Examples of such settings include sparse optimization problems, such as the LASSO, where $\phi(x) = \|Ax - b\|_2^2$, with $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $g(x) = \lambda\|x\|_1$, with $\lambda \geq 0$ and $h(x) = \frac{1}{2}\|x\|^2$. This setting also includes constrained optimization problems, where g is the indicator function of a closed convex set. We see in numerical experiments that restart schemes applied in these two settings lead to significant performance improvements.

To apply our analysis of restart schemes we need two things: an accelerated algorithm that tackles such setting and an appropriate notion of sharpness. We first introduce the notion of relative error bound.

Definition 2.5.1. *A convex function f is called relatively sharp with respect to a strictly convex function h on a set $K \subset \mathbf{dom}(f)$ iff there exist $r \geq 1$, $\mu > 0$ such that*

$$\frac{\mu}{r} D_h(x, X^*)^{\frac{r}{2}} \leq f(x) - f^* \quad \textit{for any } x \in K \qquad \text{(Relative Sharp)}$$

where $D_h(x, X^*) = \min_{x^*\in X^*} D_h(x, x^*)$ and D_h is the Bregman divergence associated to h.

If $h = \frac{1}{2}\|x\|_2^2$ we recover the definition of sharpness in the Euclidean setting (with slightly modified constants). This assumption is as generic as our first one in (Sharp) as it is satisfied if f and h are subanalytic [Bierstone and Milman, 1988, Th. 6.4].

The universal gradient is then the candidate in this setting. Given a target accuracy ε and an initial point x_0, it outputs, after t iterations, a point $x = \mathcal{U}(x_0, \varepsilon, t)$ such that

$$f(x) - f^* \leq \frac{\varepsilon}{2} + \frac{cL^{\frac{2}{s}} D_h(x_0, X^*)}{\varepsilon^{\frac{2}{s}} t^{\frac{2\rho}{s}}} \frac{\varepsilon}{2},$$

where $c = 16$ here. All our previous results can then directly be transposed to the setting here, as their proofs rely only on this convergence bound. We restate them in this setting below. First if ϕ is known to be smooth ($s = 2$) the universal fast gradient algorithm simplifies as the accelerated gradient algorithm (see Appendix A.2) and the next Corollary generalizes Proposition 2.2.2.

Corollary 2.5.2. *Let ϕ, g, h defining the composite problem (Composite) described above with ϕ satisfying (Generic Smooth) with parameters $(2, L)$ and $f = \phi + g$ satisfies the (Relative Sharp) condition with respect to h with parameters (r, μ) on a set K. Assume that we are given $x_0 \in \mathbb{R}^n$ such that $\{x |\, f(x) \leq f(x_0)\} \subset K$. Run (RESTART) from x_0 with iteration schedule $t_k = C^*_{\kappa,\tau} e^{\tau k}$, for $k = 1, \dots, R$, where*

$$C^*_{\kappa,\tau} \triangleq e^{1-\tau} (c\kappa)^{\frac{1}{2}} (f(x_0) - f^*)^{-\frac{\tau}{2}},$$

with κ and τ defined in (2.2) and $c = 16e^{2/e}$. The precision reached at the last point $\hat{x}$ is given by,

$$f(\hat{x}) - f^* \leq \exp\left(-2e^{-1}(c\kappa)^{-\frac{1}{2}} N\right) (f(x_0) - f^*) = O\left(\exp(-\kappa^{-\frac{1}{2}} N)\right), \quad \text{when } \tau = 0,$$

while,

$$f(\hat{x}) - f^* \leq \frac{f(x_0) - f^*}{\left(\tau e^{-1} (f(x_0) - f^*)^{\frac{\tau}{2}} (c\kappa)^{-\frac{1}{2}} N + 1\right)^{\frac{2}{\tau}}} = O\left(\kappa^{\frac{1}{\tau}} N^{-\frac{2}{\tau}}\right), \quad \text{when } \tau > 0,$$

where $N = \sum_{k=1}^{R} t_k$ is the total number of iterations.

For general convex functions, the following Corollary generalizes Proposition 2.3.1.

Corollary 2.5.3. *Let ϕ, g, h defining the composite problem (Composite) described above with ϕ satisfying (Generic Smooth) with parameters (s, L) on a set J and $f = \phi + g$ satisfying (Relative Sharp) with respect to h with parameters (r, μ) on a set K. Given $x_0 \in \mathbb{R}^n$ assume that $\{x |\, f(x) \leq f(x_0)\} \subset Q \cap K$. Run (Universal RESTART) scheme from x_0 for given $\varepsilon_0 \geq f(x_0) - f^*$,*

$$\gamma = \rho, \qquad t_k = C^*_{\kappa,\tau,\rho} e^{\tau k}, \quad \text{where} \quad C^*_{\kappa,\tau,\rho} \triangleq e^{1-\tau} (c\kappa)^{\frac{s}{2\rho}} \varepsilon_0^{-\frac{\tau}{\rho}}$$

where ρ is defined in (2.18), κ and τ are defined in (2.2) and $c = 16e^{2/e}$. The precision

reached at the last point $\hat{x}$ is given by,

$$f(\hat{x}) - f^* \;\leq\; \exp\left(-\rho e^{-1}(c\kappa)^{-\frac{s}{2\rho}} N\right)\varepsilon_0 \;=\; O\left(\exp(-\kappa^{-\frac{s}{2\rho}} N)\right), \quad \textit{when } \tau = 0,$$

while,

$$f(\hat{x}) - f^* \;\leq\; \frac{\varepsilon_0}{\left(\tau e^{-1}(c\kappa)^{-\frac{s}{2\rho}} \varepsilon_0^{\frac{\tau}{\rho}} N + 1\right)^{\frac{\rho}{\tau}}} \;=\; O\left(\kappa^{\frac{s}{2\tau}} N^{-\frac{\rho}{\tau}}\right), \quad \textit{when } \tau > 0,$$

where $N = \sum_{k=1}^{R} t_k$ is total number of iterations.

The results regarding adaptive schemes and those with termination criterion generalize similarly under the relative sharpness assumption.

2.6 Numerical results

We illustrate our results by testing our adaptive restart methods, denoted *Adap* and *Crit* (ε-Restart), introduced respectively in Sections 2.2.2 and 2.4 on several problems and compare them against simple gradient descent (*Grad*), accelerated gradient methods (*Acc*), and the restart heuristic enforcing monotonicity (*Mono* in [O'Donoghue and Candes, 2015]). For *Adap* we plot the convergence of the best method found by grid search to compare with the restart heuristic. This implicitly assumes that the grid search is run in parallel with enough servers. For *Crit* we use the optimal f^* found by another solver. This gives an overview of its performance in order to potentially approximate it along the iterations as with Polyak steps [Poljak, 1987]. All restart schemes are using the accelerated gradient with backtracking line search detailed in the Supplementary Material, with large dots representing restart iterations.

In Figures 2-1, we solve classification problems with various losses on the UCI *Sonar* data set [Asuncion and Newman, 2007] with $(n, d) = (208, 60)$. For least square loss on sonar data set, we observe much faster convergence of the restart schemes compared to the accelerated method. These results were already observed by O'Donoghue and Candes [2015]. For logistic loss, we observe that restart does not provide much improvement. For hinge loss, we regularized by a squared norm and optimize the dual, which means solving a quadratic problem with box constraints. We observe here that the scheduled restart scheme converges much faster, while restart heuristics may be activated too late. We observe similar results for the LASSO problem. In general *Crit* ensures acceleration rates consistent with the theory but *Adap* exhibits more consistent behavior. This highlights the benefits of a Hölderian error bound assumption for these last two problems. Precisely quantifying sharpness of the function from data/problem structure is a key open problem.

To account for the grid search effort, in Figure 2-2, we multiplied the number of iterations made by the *Adap* method by the size of the grid. This is for the LASSO problem on *Sonar* data set with a grid step size of 4. This shows that the benefits

of the restart schemes make the grid search effort acceptable both on paper and in practice. More clever grid search strategies for scheduled restarts run in parallel would even reduce the grid search effort.

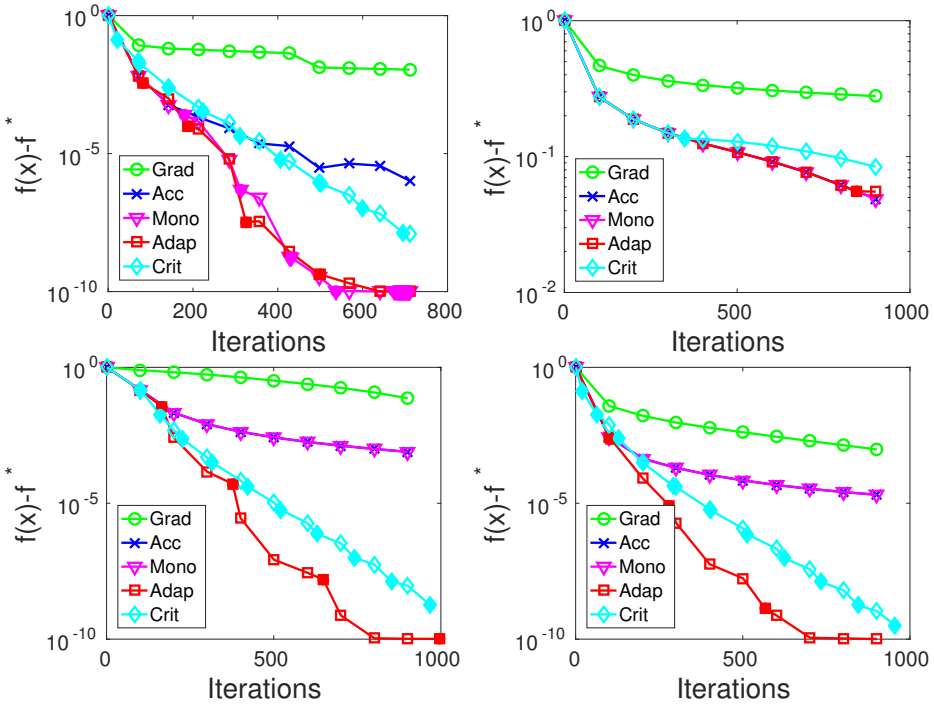

Figure 2-1 – Sonar data set. From top to bottom and left to right: least square loss, logistic loss, dual SVM problem and LASSO. We use adaptive restarts (Adap), gradient descent (Grad), accelerated gradient (Acc) and restart heuristic enforcing monotonicity (Mono). Large dots represent the restart iterations. Regularization parameters for dual SVM and LASSO were set to one.

2.7 Conclusion

We have shown that Hölderian error bounds that generically describe convex functions can be optimally treated by restart schemes of fast convex optimization algorithms for smooth or non-smooth functions. These restarts are shown to be robust to misspecification of the parameters of the error bound as a log-scale grid search on these parameters offer near-optimal rates for smooth convex functions. Schedules are however still a drawback of our approach as they do not adapt to local geometries of the function. Error bounds can indeed be formulated at the distance to a sublevel set of the function instead of the set of minimizers. Our restart with criterion brings a partial resolution of the question when f^* is known. Recently Fercoq and Qu [2017] developed generic adaptive schemes for quadratic local error bounds that give theoretical guarantees of acceleration for ℓ_1 regularized or constrained problems by restart

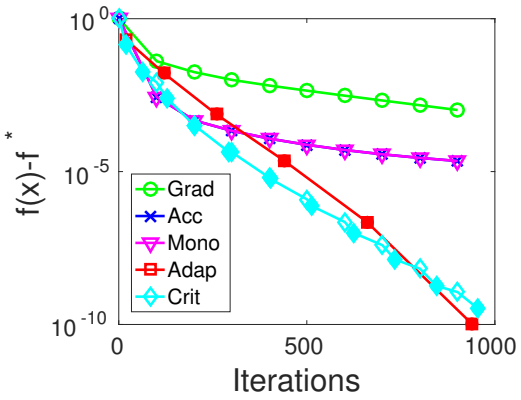

Figure 2-2 – Comparison of the methods for the LASSO problem on Sonar dataset where number of iterations of the Adaptive method is multiplied by the size of the grid. Grid search step size is set to 4.

schemes with criterion. Extending such results to sharp non-smooth problems such as sparse recovery problems presented in next chapter is a potential future direction of research.

Appendix

2.A Rounding issues

We presented convergence bounds for real sequences of iterate counts $(t_k)_{k=1}^{\infty}$ but in practice these are integer sequences. The following Lemma details the convergence of our schemes for an approximate choice $\tilde{t}_k = \lceil t_k \rceil$.

Lemma 2.A.1. *Let x_k be a sequence whose k^{th} iterate is generated from previous one by an algorithm that needs t_k iterations and denote $N = \sum_{k=1}^{R} t_k$ the total number of iterations to output a point $\hat{x} = x_R$. Suppose setting*

$$t_k = \lceil Ce^{\alpha k} \rceil, \quad k = 1, \dots, R$$

for some $C > 0$ and $\alpha \geq 0$ ensures that objective values $f(x_k)$ converge linearly, i.e.

$$f(x_k) - f^* \leq \nu e^{-\gamma k}, \tag{2.22}$$

for all $k \geq 0$ with $\nu \geq 0$ and $\gamma \geq 0$. Then precision at the output is given by,

$$f(\hat{x}) - f^* \leq \nu \exp(-\gamma N/(C+1)), \quad \text{when } \alpha = 0,$$

and

$$f(\hat{x}) - f^* \leq \frac{\nu}{(\alpha e^{-\alpha} C^{-1} N' + 1)^{\frac{\gamma}{\alpha}}}, \quad \text{when } \alpha > 0,$$

where $N' = N - \frac{\log\left((e^{\alpha}-1)e^{-\alpha}C^{-1}N+1\right)}{\alpha}$.

Proof. At the R^{th} point generated, $N = \sum_{k=1}^{R} t_k$. If $t_k = \lceil C \rceil$, define $\varepsilon = \lceil C \rceil - C$ such that $0 \leq \varepsilon < 1$. Then $N = R(C + \varepsilon)$, injecting it in (2.22) at the R^{th} point,

$$f(\hat{x}) - f^* \leq \nu e^{-\gamma \frac{N}{C+\varepsilon}} \leq \nu e^{-\gamma \frac{N}{C+1}}.$$

Now, if $t_k = \lceil Ce^{\alpha k} \rceil$, define $\varepsilon_k = \lceil Ce^{\alpha k} \rceil - Ce^{\alpha k}$, such that $0 \leq \varepsilon_k < 1$. On one hand

$$N \geq \sum_{k=1}^{R} Ce^{\alpha k},$$

such that

$$R \leq \frac{\log\left((e^{\alpha} - 1)e^{-\alpha}C^{-1}N + 1\right)}{\alpha}.$$

On the other hand,

$$\begin{aligned} N = \sum_{k=1}^{R} t_k &= \frac{Ce^{\alpha}}{e^{\alpha}-1}(e^{\alpha R}-1) + \sum_{k=1}^{R} \varepsilon_k \\ &\leq \frac{Ce^{\alpha}}{e^{\alpha}-1}(e^{\alpha R}-1) + R \\ &\leq \frac{Ce^{\alpha}}{e^{\alpha}-1}(e^{\alpha R}-1) + \frac{\log\left((e^{\alpha}-1)e^{-\alpha}C^{-1}N+1\right)}{\alpha}, \end{aligned}$$

such that

$$R \geq \frac{\log\left(\alpha e^{-\alpha}C^{-1}N'+1\right)}{\alpha}.$$

Injecting it in (2.22) at the R^{th} point the result follows. ■

Chapter 3

A brief introduction to sparse problems

Sparsity has been studied early on by statisticians to select relevant features in data fitting problem, see Guyon and Elisseeff [2003] for a review. A whole field, namely compressed sensing, has then been dedicated to its analysis as it provides new coding-decoding procedures that break Shannon compression rate barrier, see Candès and Wakin [2008] for an introduction. Here we briefly recall models that tackle sparsity assumptions and their structure for which geometrical analysis provides insights on their optimization process.

3.1 Original sparse problems

The term sparsity has been extensively used to qualify diverse structures of variables that compress their information. Originally a vector is said to be sparse if it has few non-zero coefficients. Formally, a vector $x \in \mathbb{R}^d$ is s-sparse if its support $\mathrm{Supp}(x) = \{i \in [\![1, d]\!] : x_i \neq 0\}$ has cardinality at most s. Sparsity assumption can be used for example in linear regression problems. Assume that we are given n noisy observations $b_1, \dots, b_n$ of a s-sparse vector $x_* \in \mathbb{R}^d$ by data points $a_1, \dots, a_n \in \mathbb{R}^d$ such that

$$b_i = a_i^T x_i + \eta_i, \quad \text{for every } i \in [\![1, d]\!],$$

where η_i are bounded i.i.d. noise. Simple least-square estimation can retrieve x^* if the number of samples n is larger than the number of features d. However as it does not exploit the sparsity assumption, estimated solution will be dense and some coefficients only capture noise signal. Moreover if the number of samples is strictly less than the number of features, least-square estimation simply fails.

3.1.1 Models

Several models have thus been developed to exploit sparsity assumption and recover the original signal even if $n < d$. When observations are not perturbed by noise a straight strategy to recover original signal is to minimize the number of non-zero

coefficients of vectors satisfying the observations. However this problem is NP-hard [Natarajan, 1995] so one rather seeks to solve its convex relaxation called Basis Pursuit [Chen et al., 2001] that reads

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & Ax = b \end{array} \tag{3.1}$$

in variable $x \in \mathbb{R}^d$, where $A = (a_1, \ldots, a_n)^T \in \mathbb{R}^{n \times d}$ is the matrix of data points and $b = (b_1, \ldots, b_n) \in \mathbb{R}^n$ is the vector of observations. To account for noise, a robust version of (3.1) reads

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & \|Ax - b\|_2 \leq \delta \|A\|_2, \end{array} \tag{3.2}$$

in variable $x \in \mathbb{R}^d$, where δ is an estimation of the level of noise and $\|A\|_2$ is the spectral norm of the observation matrix A.

Another approach to retrieve the original signal is to minimize a data fitting term under sparsity constraints. As these latter are not convex, one consider the relaxed versions that constrains the ℓ_1 norm of the variable. Its penalized version is well known as the LASSO [Tibshirani, 1996] problem that reads

$$\text{minimize} \quad \tfrac{1}{2}\|Ax - b\|_2^2 + \lambda \|x\|_1 \tag{3.3}$$

in variable $x \in \mathbb{R}^d$, where $\lambda \geq 0$ is a regularization parameter. Finally by looking at the optimal conditions of the LASSO problem, another estimator called Dantzig selector [Candes and Tao, 2007] was developed.

3.1.2 Recovery performance

The main question that drew interest of the compressed sensing community was to certify that, by solving these problems, one can retrieve the original vector at least approximatively, i.e. decode it from the observations. Precisely, they were interested in the number of observations needed to recover the signal and the robustness to noise of these procedures. Naturally the answer depends on properties of the observation matrix A, i.e. how the original vector was encoded by linear observations.

In their seminal works Candès et al. [2006]; Candes and Tao [2006]; Donoho [2006] show that for an original s-sparse vector, $O(s \log d)$ random observations of it are sufficient to decode it by solving one of these problems. To this end, they introduce the restricted isometric property defined below.

Definition 3.1.1. *A matrix $A \in \mathbb{R}^{n \times d}$ satisfies the Restricted Isometric Property (RIP) at level s, if there exists δ_s such that for any submatrix $A_s \in \mathbb{R}^{n \times s}$ and any s-sparse vector x,*

$$(1 - \delta_s)\|x\|_2^2 \leq \|A_s x\|_2^2 \leq (1 + \delta_s)\|x\|_2^2.$$

Candès et al. [2006]; Candes and Tao [2006] then proved that the RIP at level s with sufficiently small parameter δ_s ensures recovery of s-sparse signals and that if

the matrix of observations A is drawn from a Gaussian random distribution then the RIP is satisfied.

Further assumptions on the observation matrix A were then defined to ensure recovery. Donoho and Huo [2001]; Feuer and Nemirovski [2003]; Cohen et al. [2009] notably introduced the null space property defined below.

Definition 3.1.2. *A matrix A satisfies the Null Space Property (NSP) at order s with constant $\alpha \geq 1$ if for any support $S \subset \{1, \dots p\}$ of cardinality at most s and any $z \in \mathrm{Null}(A) \setminus \{0\}$,*

$$\alpha \|z_S\|_1 < \|z_{S^c}\|_1,$$

where $\mathrm{Null}(A)$ is the null space of A, and $z_S \in \mathbb{R}^p$ denotes the vector obtained by zeroing all coefficients of z that are not in S.

NSP provides sufficient and necessary conditions for exact recovery of signals in the absence of noise when solving problem(3.1). For LASSO problem (3.3) or Dantzig selector, numerous properties were defined, Van De Geer et al. [2009] summarized and linked them. Minimal conically restricted singular values defined by Bickel et al. [2009] that read for a matrix A,

$$\kappa(s, \alpha) = \inf_{\substack{S \in [\![1,p]\!] \\ \mathbf{Card}(S) \leq s}} \inf_{\substack{\|x_{S^c}\|_1 \leq \alpha \|x_S\|_1 \\ \|x_S\|_2 = 1}} \|Ax\|_2$$

appeared to be the essential quantity that control exact and robust recovery. These assumptions were linked early to Kolmogorov width [Kashin and Temlyakov, 2007] or Gaussian widths [Chandrasekaran et al., 2012; Raskutti et al., 2010] of sections of unit balls which allow a simpler and more geometrical approach to these problems. Overall, these works provide a fine description of the geometry of the problem that can be used for optimization purposes.

3.2 Optimization procedures

On an optimization point of view problems (3.1) and (3.2) on one side and problem (3.3) have different structures and are therefore tackled by different procedures that we present now.

3.2.1 Basis pursuits

As the ℓ_1 norm is non-smooth, problems (3.1) and (3.2) belong to the class of non-smooth convex problems and shall have complexity $O(1/\sqrt{\varepsilon})$. However analysis does not take into account the structure of the problem, namely that the ℓ_1 norm can be written as a maximum of linear functions. In his seminal paper, Nesterov [2005] presented how this additional information can be exploited to get an approximate solution in $O(1/\varepsilon)$ iterations by smoothing the objective function and applying his accelerated algorithm for smooth convex optimization. This remark was exploited by Becker, Bobin and Candès [2011] on problems (3.1) and (3.2). They also provide

restarting heuristics of their algorithm, called continuation steps, that show remarkable performance but they do not explain this phenomenon.

Łojasiewicz inequality can then help. While the ℓ_1 norm is clearly not strongly convex, it is sharp, i.e. it satisfies

$$\gamma d(x, X^*) \leq f(x) - f^*, \quad \text{for every } x \in \mathbb{R}^d$$

where $f(x) = \|x\|_1$, $f^* = 0$, $X^* = 0$, $d(x, X^*)$ is the ℓ_1 distance of x to the minimizer and $\gamma = 1$. This property generalizes to problem (3.1) and can be exploited to produce linearly convergent restart schemes as shown in Chapter 4. The sharpness constant depends then on the data A that defines the set of constraints. Interestingly, sharpness can be estimated by the same conic properties of the problem that control recovery performance. A finer analysis in Chapter 4 show that a single quantity controls then both statistical and computational performance.

3.2.2 LASSO

The LASSO problem (3.3) drew more attention of the optimization community for its applicability in prediction problems. Accelerated methods for smooth convex composite problems apply in this setting, they were first developed by Beck and Teboulle [2009] with FISTA algorithm. As the ℓ_1 norm is decomposable, coordinate accelerated methods were also proposed [Fercoq and Richtárik, 2015; Lee and Sidford, 2013; Nesterov, 2012]. Some specialized schemes discard features that are certified to be zero along the optimization process see Fercoq et al. [2015] and references herein. Finally homotopy methods such LARS [Efron et al., 2004] build the regularization path of the method, i.e. the curve of solutions for varying regularization parameters.

In the undetermined case, where $n < d$, problem (3.3) is convex but not strongly convex. However here again restart schemes were shown to outperform simple accelerated methods [O'donoghue and Candes, 2015]. Further properties of the problem were explored to prove linear convergence of a simple gradient descent [Yen et al., 2014; Agarwal et al., 2010]. A closer inspection of the structure highlighted that LASSO problem satisfies the Łojasiewicz inequality with exponent $r = 2$ [Bolte et al., 2015; Zhang, 2017], which explained once more why restart schemes accelerate standard methods and allow theoretical guarantees as done by Fercoq and Qu [2016, 2017].

3.3 Generalized sparse structure

Original sparsity is one assumption among others that present how some variables can be compressed. For example if the variable is a matrix rather than a vector, compression can mean low rank of the matrix. Several point of views were taken to ensure such compressed structures, loosely called sparse structures. A first approach is to consider these structures as the description of variables by a finite set of "atoms" [Chandrasekaran et al., 2012]. The norm induced by the convex hull of these atoms can then be used to build specific estimators. Their geometrical definition has then

been used to analyze recovery performance of problems of the form (3.1), (3.2) or (3.3) [Raskutti et al., 2010].

Combinatorial penalties can also be used to induce sparse structures on the variables. Original sparsity for example is the cardinality of the support of the variable. It was generalized by using submodular functions on sets defined by a vector, like its support [Bach et al., 2013]. Their continuous extensions can then be used as regularizers for machine learning problems [Bach et al., 2012]. Furthermore by identifying the underlying combinatorial problem, this approach provides practical operators for the resulting regularizer. Notice that generally the support function of the symmetric base polytope of the support function is the dual norm of an atomic norm, such that both point of views merge.

Finally several frameworks attempt to unify sparse structures under generic assumptions to identify their statistical properties [Negahban et al., 2009]. Recovery performances of corresponding problems require then generalization of the previous properties as done e.g. by Juditsky et al. [2014]; Oymak and Hassibi [2010].

Chapter 4

On computational and statistical performances of sparse recovery problems

Chapter Abstract

In the past years, sparse recovery problems have received a lot of attention from various perspectives. On one side, an extensive literature explores attainable statistical performance, that is, the possibility to retrieve the underlying original signal, depending on the number of observations or the noise level. On the other side, a long list of algorithms now solve these problems at a low computational cost. Here, we provide evidence that these two aspects are intimately related. Loosely stated, the more accurate the recovery is, the faster it is to compute. As a first step, we develop a linearly convergent restart scheme for exact recovery whose rate is controlled by the recovery threshold of the problem. Then, we analyze the underlying conic geometry of sparse recovery problems. On the statistical side, minimal conically-restricted singular values of the observation matrix control robust recovery performance. On the computational side, Renegar's condition number, generalizing classical condition numbers in linear algebra to conic systems, is shown to control computational complexity of optimality certificates for exact recovery and the restart scheme we present. Numerical experiments illustrate its impact for several other classical algorithms. By observing that the worst-case value of this algorithmic complexity measure taken over all signals matches the minimal conically-restricted singular value, we argue that, in these problems, a single parameter directly controls computational complexity and statistical recovery performance

Introduction

Sparse recovery problems have received a lot of attention from various perspectives. On one side, an extensive literature explores the limits of recovery performance. On the other side, a long list of algorithms now solve these problems very efficiently. Early on, it was noticed empirically by e.g. Donoho and Tsaig [2008], that recovery problems which are easier to solve from a statistical point of view (i.e., where more samples are available), are also easier to solve numerically. Here, we show that these two aspects are indeed intimately related.

Recovery problems consist in retrieving a signal x^*, lying in some Euclidean space E, given linear observations. If the signal is "sparse", namely if it can be efficiently compressed, a common approach is to minimize the corresponding sparsity inducing norm $\|\cdot\|$ (e.g. the ℓ_1 norm in classical sparse recovery). The exact sparse recovery problem then reads

$$\begin{array}{ll} \text{minimize} & \|x\| \\ \text{subject to} & A(x) = b, \end{array} \tag{4.1}$$

in the variable $x \in E$, where A is a linear operator on E and $b = A(x^*)$ is the vector of observations. If the observations are affected by noise a robust version of this problem is written as

$$\begin{array}{ll} \text{minimize} & \|x\| \\ \text{subject to} & \|A(x) - b\|_2 \leq \varepsilon, \end{array}$$

in the variable $x \in E$, where $\|\cdot\|_2$ is the Euclidean norm and $\varepsilon > 0$ is a tolerance to noise. In penalized form, this is

$$\text{minimize} \quad \|x\| + \lambda \|A(x) - b\|_2^2$$

in the variable $x \in E$ where $\lambda > 0$ is a penalization parameter. This last problem is known as the LASSO [Tibshirani, 1996] in the ℓ_1 case.

When x^* has no more than s non zero values, Donoho and Tanner [2005] and Candes and Tao [2006] have shown that, for certain linear operators A, $O(s \log p)$ observations suffice for stable recovery of x^* by solving the exact formulation (4.1) using the ℓ_1 norm (a linear program). These results have then been generalized to many other recovery problems with various assumptions on signal structure (e.g., where x is a block-sparse vector, a low-rank matrix, etc.) and corresponding convex relaxations were developed in those cases (see e.g. Chandrasekaran et al. [2012] and references therein). Recovery performance is often measured in terms of the number of samples required to guarantee exact or robust recovery given a level of noise.

On the computational side, many algorithms were developed to solve these problems at scale. Besides specialized methods such as LARS [Efron et al., 2004], FISTA [Beck and Teboulle, 2009] and NESTA [Becker, Bobin and Candès, 2011], solvers use accelerated gradient methods to solve robust recovery problems, with efficient and flexible implementations covering a wide range of compressed sensing instances developed by e.g. Becker, Candès and Grant [2011]. Recently, linear convergence results have been obtained for the LASSO [Agarwal et al., 2010; Yen et al., 2014;

Zhou et al., 2015] using variants of the classical strong convexity assumption. Some restart schemes have also been developed in e.g. [O'Donoghue and Candes, 2015; Su et al., 2014; Giselsson and Boyd, 2014] while Fercoq and Qu [2016] showed that generic restart schemes can offer linear convergence given a rough estimate of the behavior of the function around its minimizers.

As mentioned above, Donoho and Tsaig [2008] was one of the first reference to connect statistical and computational performance in this case, showing empirically that recovery problems which are easier to solve from a statistical point of view (i.e., where more samples are available), are also easier to solve numerically (using homotopy methods). More recently, Chandrasekaran and Jordan [2013]; Amelunxen et al. [2014] studied computational and statistical trade offs for increasingly tight convex relaxations of shrinkage estimators. They show that recovery performance is directly linked to the Gaussian squared-complexity of the tangent cone with respect to the constraint set and study the complexity of several convex relaxations. In [Chandrasekaran and Jordan, 2013; Amelunxen et al., 2014] however, the structure of the convex relaxation is varying and affecting both complexity and recovery performance, while in [Donoho and Tsaig, 2008] and in what follows, the structure of the relaxation is fixed, but the data (i.e. the observation matrix A) varies.

Here, as a first step, we study the exact recovery case and show that the null space property introduced by Cohen et al. [2009] can be seen as a measure of sharpness on the optimum of the sparse recovery problem. On one hand this allows us to develop linearly convergent restart schemes whose rate depends on this sharpness. On the other hand we recall how the null space property is linked to the recovery threshold of the sensing operator A for random designs, thus producing a clear link between statistical and computational performance.

We then analyze the underlying conic geometry of recovery problems. Robust recovery performance is controlled by a minimal conically restricted singular value. We recall Renegar's condition number and show how it affects the computational complexity of optimality certificates for exact recovery and the linear convergence rate of restart schemes. By observing that the minimal conically restricted singular value matches the worst case value of Renegar's condition number on sparse signals, we provide further evidence that a single quantity controls both computational and statistical aspects of recovery problems. Numerical experiments illustrate its impact on various classical algorithms for sparse recovery.

The first two sections focus on the ℓ_1 case for simplicity. We generalize our results to non-overlapping group norms and the nuclear norm in a third section.

Notations

For a given integer $p \geq 1$, $[\![1, p]\!]$ denotes the set of integers between 1 and p. For a given subset $S \subset [\![1, p]\!]$, we denote $S^c = [\![1, p]\!] \setminus S$ its complementary and $\mathbf{Card}(S)$ its cardinality. For a given vector $x \in \mathbb{R}^p$, we denote $\mathrm{Supp}(x) = \{i \in [\![1, p]\!] : x_i \neq 0\}$ the support of x, $\|x\|_0 = \mathbf{Card}(\mathrm{Supp}(x))$ its sparsity and $\|x\|_p$ its p-norm. For a given vector x and integer subset $S \subset [\![1, p]\!]$, $x_S \in \mathbb{R}^p$ denotes the vector obtained by zeroing all coefficients of x that are not in S. For a given linear operator or matrix A,

we denote Null(A) its null space, $\Im(A)$ its range, and $\|X\|_2$ its operator norm with respect to the Euclidean norm (for matrices this is the spectral norm). The identity operator is denoted $\mathbf{I}$. In a linear topological space E we denote $\mathbf{int}(F)$ the interior of $F \subset E$. Finally for a given real a, we denote $\lceil a \rceil$ the smallest integer larger than or equal to a and $\lfloor a \rfloor$ the largest integer smaller than or equal to a.

4.1 Recovery performance and linear convergent restart scheme for exact recovery

In this section and the following one, we discuss sparse recovery problems using the ℓ_1 norm. Given a matrix $A \in \mathbb{R}^{n \times p}$ and observations $b = Ax^*$ on a signal $x^* \in \mathbb{R}^p$, recovery is performed by solving the ℓ_1 minimization program

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & Ax = b \end{array} \qquad (\ell_1 \text{ recovery})$$

in the variable $x \in \mathbb{R}^p$.

In what follows, we show that the Null Space Property condition (recalled below) can be seen as measure of sharpness for ℓ_1-recovery of a sparse signals x^*, with

$$\|x\|_1 - \|x^*\|_1 > \gamma \|x - x^*\|_1 \qquad \text{(Sharp)}$$

for any $x \neq x^*$ such that $Ax = b$, and some $0 \leq \gamma < 1$. This first ensures that x^* is the unique minimizer of problem (ℓ_1 recovery) but also has important computational implications. It allows us to produce linear convergent restart schemes whose rates depend on sharpness. By connecting null space property to recovery threshold for random observation matrices, we thus get a direct link between computational and statistical aspects of sparse recovery problems.

4.1.1 Null space property & sharpness for exact recovery

Although the definition of null space property appeared in earlier work [Donoho and Huo, 2001; Feuer and Nemirovski, 2003] the terminology of restricted null space is due to Cohen et al. [2009]. The following definition differs slightly from the original one in order to relate it to intrinsic geometric properties of the problem in Section 4.2.

Definition 4.1.1. (Null Space Property) *The matrix A satisfies the Null Space Property (NSP)* on support $S \subset [\![1, p]\!]$ *with constant $\alpha \geq 1$ if for any $z \in \text{Null}(A) \backslash \{0\}$,*

$$\alpha \|z_S\|_1 < \|z_{S^c}\|_1. \qquad \text{(NSP)}$$

The matrix A satisfies the Null Space Property at order s *with constant $\alpha \geq 1$ if it satisfies it on every support S of cardinality at most s.*

The Null Space Property is a necessary and sufficient condition for the convex program (ℓ_1 recovery) to recover all signals up to some sparsity threshold. Necessity

will follow from results recalled in Section 4.2.2. We detail sufficiency of (NSP) using sharpness in the following proposition.

Proposition 4.1.2. *Given a coding matrix $A \in \mathbb{R}^{n\times p}$ satisfying* (NSP) *at order s with constant $\alpha \geq 1$, if the original signal x^* is s-sparse, then for any $x \in \mathbb{R}^p$ satisfying $Ax = b$, $x \neq x^*$, we have*

$$\|x\|_1 - \|x^*\|_1 > \frac{\alpha - 1}{\alpha + 1}\|x - x^*\|_1.$$

This implies signal recovery, i.e. optimality of x^* for (ℓ_1 recovery)*, and the sharpness bound* (Sharp) *with $\gamma = \frac{\alpha-1}{\alpha+1}$.*

Proof. The proof follows the one in Cohen et al. [2009, Theorem 4.4]. Let $S = \mathrm{supp}(x^*)$, with $\mathbf{Card}(S) \leq s$, and let $x \neq x^*$ such that $Ax = b$, so $z = x - x^* \neq 0$ satisfies $Az = 0$. Then

$$\begin{aligned}\|x\|_1 &= \|x^*_S + z_S\|_1 + \|z_{S^c}\|_1 \\ &\geq \|x^*_S\|_1 - \|z_S\|_1 + \|z_{S^c}\|_1 \\ &= \|x^*\|_1 + \|z\|_1 - 2\|z_S\|_1.\end{aligned}$$

Now as A satisfies (NSP) on support S,

$$\|z\|_1 = \|z_S\|_1 + \|z_{S^c}\|_1 > (1 + \alpha)\|z_S\|_1$$

hence

$$\|x\|_1 - \|x^*\|_1 > \frac{\alpha - 1}{\alpha + 1}\|z\|_1 = \frac{\alpha - 1}{\alpha + 1}\|x - x^*\|_1.$$

As $\alpha \geq 1$, this implies that x^* is the solution of program (ℓ_1 recovery) and the corresponding sharpness bound. ∎

Sharpness is a crucial property for optimization problems that can be exploited to accelerate the performance of classical optimization algorithms [Nemirovskii and Nesterov, 1985; Roulet and d'Aspremont, 2017]. Before that we remark that it is in fact equivalent to (NSP) at order s.

Proposition 4.1.3. *Given a matrix $A \in \mathbb{R}^{n\times p}$ such that problem* (ℓ_1 recovery) *is sharp on every s-sparse signal x^*, i.e. there exists $0 \leq \gamma < 1$ such that*

$$\|x\|_1 - \|x^*\|_1 > \gamma\|x - x^*\|_1,$$

for any $x \neq x^*$ such that $Ax = Ax^*$. Then, A satisfies (NSP) *at order s with constant $\alpha = \frac{1+\gamma}{1-\gamma} \geq 1$.*

Proof. Let $S \subset [\![1, p]\!]$ with $\mathbf{Card}(S) \leq s$ and $z \in \mathrm{Null}(A)$, $z \neq 0$, such that $Az_S = -Az_{S^c}$ and $z_S \neq -z_{S^c}$. Using sharpness of problem (ℓ_1 recovery) with $x^* = z_S$,

and $x = -z_{S^c}$, we get

$$\|z_{S^c}\|_1 - \|z_S\|_1 > \gamma\|z\|_1 = \gamma\|z_S\|_1 + \gamma\|z_{S^c}\|_1.$$

Rearranging terms and using $\gamma < 1$, this reads

$$\|z_{S^c}\|_1 > \frac{1+\gamma}{1-\gamma}\|z_S\|_1,$$

which is (NSP) on support S with the announced constant. As S was taken arbitrarily, this means (NSP) holds at order s. ■

4.1.2 Restarting first-order methods

In this section, we seek to solve the recovery problem (ℓ_1 recovery) and exploit the sharpness bound (Sharp). The NESTA algorithm [Becker, Bobin and Candès, 2011] uses the smoothing argument of Nesterov [2013*a*] to solve (ℓ_1 recovery). In practice, this means using the optimal algorithm of Nesterov [1983] to minimize

$$f_\varepsilon(x) \triangleq \sup_{\|u\|_\infty \leq 1} \left\{ u^T x - \frac{\varepsilon}{2p}\|u\|_2^2 \right\}$$

for some $\varepsilon > 0$, which approximates the ℓ_1 norm uniformly up to $\varepsilon/2$. This is the classical Huber function, which has a Lipschitz continuous gradient with constant equal to p/ε. Overall given an accuracy ε and a starting point x_0 this method outputs after t iterations a point $x = \mathcal{A}(x_0, \varepsilon, t)$ such that

$$\|x\|_1 - \|\hat{x}\|_1 \leq \frac{2p\|x_0 - \hat{x}\|_2^2}{\varepsilon t^2} + \frac{\varepsilon}{2}, \tag{4.2}$$

for any $\hat{x}$ solution of problem (ℓ_1 recovery). Now if the sharpness bound is satisfied, restarting this method, as described in the (Restart) scheme presented below, accelerates its convergence.

Algorithm 4 Restart Scheme **(Restart)**

Input: Initial point $y_0 \in \mathbb{R}^p$, initial gap $\varepsilon_0 \geq \|y_0\|_1 - \|\hat{x}\|_1$, decreasing factor ρ, restart clock t

For $k = 1 \ldots, K$ compute

$$\varepsilon_k = \rho\varepsilon_{k-1}, \qquad y_k = \mathcal{A}(y_{k-1}, \varepsilon_k, t) \tag{Restart}$$

Output: A point $\hat{y} = y_K$ approximately solving (ℓ_1 recovery).

Optimal restart scheme

We begin by analyzing an optimal restart scheme assuming the sharpness constant is known. We use a non-integer clock to highlight its dependency to the sharpness. Naturally clock and number of restarts must be integer but this does not affect much bounds as detailed in Appendix 4.A. The next proposition shows that algorithm $\mathcal{A}$ needs a constant number of iterations to decrease the gap by a constant factor, which means restart leads to linear convergence.

Proposition 4.1.4. *Given a coding matrix $A \in \mathbb{R}^{n\times p}$ and a signal $x^* \in \mathbb{R}^p$ such that the sharpness bound (Sharp) is satisfied with $\gamma > 0$, i.e.*

$$\|x\|_1 - \|x^*\|_1 > \gamma\|x - x^*\|_1,$$

for any $x \neq x^*$ such that $Ax = Ax^*$, running the (Restart) scheme with $t \geq \frac{2\sqrt{p}}{\gamma\rho}$ ensures

$$\|y_k\|_1 - \|x^*\|_1 \leq \varepsilon_k, \tag{4.3}$$

at each iteration, with x^* the unique solution of problem (ℓ_1 recovery). Using optimal parameters

$$\rho^* = e^{-1} \qquad \text{and} \qquad t^* = \frac{2e\sqrt{p}}{\gamma}, \tag{4.4}$$

we get a point $\hat{y}$ such that

$$\|\hat{y}\|_1 - \|x^*\|_1 \leq \exp\left(-\frac{\gamma}{2\sqrt{p}}eN\right)\varepsilon_0. \tag{4.5}$$

after running a total of N inner iterations of Algorithm Restart with $t = t^*$ (hence N/t restarts).

Proof. By the choice of ε_0, (4.3) is satisfied for $k = 0$. Assuming it holds at iteration k, combining sharpness bound (Sharp) and complexity bound (4.2) leads to, for $x = \mathcal{A}(y_{k-1}, \varepsilon_k, t)$,

$$\begin{aligned}\|x\|_1 - \|x^*\|_1 &\leq \frac{2p(\|y_{k-1}\|_1 - \|x^*\|_1)^2}{\gamma^2\varepsilon_k t^2} + \frac{\varepsilon_k}{2} \\ &\leq \frac{4p}{\rho^2\gamma^2t^2}\frac{\varepsilon_k}{2} + \frac{\varepsilon_k}{2}.\end{aligned}$$

Therefore after $t \geq \frac{2\sqrt{p}}{\gamma\rho}$ iterations, the method has achieved the decreased accuracy ε_k which proves (4.3). The overall complexity after a total of N inner iterations, hence N/t restarts, is then

$$\|\hat{y}\|_1 - \|x^*\|_1 \leq \rho^{N/t}\varepsilon_0.$$

If γ is known, using exactly $\frac{2\sqrt{p}}{\gamma\rho}$ inner iterations at each restart leads to

$$\|\hat{y}\|_1 - \|x^*\|_1 \leq \exp\left(\frac{\gamma}{2\sqrt{p}}N\rho\log\rho\right)\varepsilon_0.$$

Optimizing in ρ yields $\rho^* = e^{-1}$, and with t^* inner iterations the complexity bound (4.5) follows. ∎

To run NESTA, $A^T A$ is assumed to be an orthogonal projector (often the case in compressed sensing applications) such that the projection on the feasible set is easy. Becker, Bobin and Candès [2011] already studied restart schemes that they called "acceleration with continuation". However their restart criterion depends on the relative variation of objective values, not on the number of iterates, and no linear convergence was proven. We further note that linear convergence of restart schemes only requires an assumption of the form

$$f(x) - f^* \geq \gamma d(x, X^*), \tag{4.6}$$

where $d(x, X^*)$ is the distance (in any norm) from x to the set of minimizers of the objective function f (here $f(x) = \|x\|_1$). This type of bound is known as Łojasiewicz's inequality, studied for example in Bolte et al. [2007] for non-smooth convex functions. Here (NSP) ensures that the set of minimizers is reduced to a singleton, the original signal.

Practical restart scheme

Several parameters are needed to run the optimal scheme above. The optimal decreasing factor is independent of the data. The initial gap ε_0 can be taken as $\|y_0\|_1$ for $Ay_0 = b$. The sharpness constant γ is for its part mostly unknown such that we cannot choose the number t^* of inner iterations a priori. However, given a budget of iterations N, a log scale grid search can be performed on the optimal restart clock to get nearly optimal rates as detailed in the following corollary (contrary to the general results in [Roulet and d'Aspremont, 2017], the sharpness exponent is known here, simplifying the parameter search).

Corollary 4.1.5. *Given a coding matrix $A \in \mathbb{R}^{n \times p}$, a signal $x^* \in \mathbb{R}^p$ such that the sharpness bound (Sharp) is satisfied with $\gamma > 0$, a budget of N iterations, run the following schemes from an initial point y_0*

$$\text{(Restart) with} \quad t = h^j, \qquad j = 1, \dots, \lfloor \log_h N \rfloor$$

with h the grid search precision. Stop restart iteration when the total number of iterations has exceeded the budget N. Then, provided that $N \geq ht^*$, where t^* is defined in (4.4), at least one of these restart schemes achieves a precision given by

$$\|\hat{y}\|_1 - \|x^*\|_1 \leq \exp\left(-\frac{\gamma}{2h\sqrt{p}} eN\right) \varepsilon_0. \tag{4.7}$$

Overall running the logarithmic grid search has a complexity $\log_h N$ times higher than running N iterations in the optimal scheme.

Proof. All schemes stop after at most $N + h^j \leq 2N$ iterations. As we assumed $N \geq ht^*$, $j = \lceil \log_h t^* \rceil \leq \log_h N$ and (Restart) has been run with $t = h^j$. Proposi-

tion 4.1.4 ensures, since $t \geq t^*$, that the output of this scheme achieves after $N' \geq N$ total iterations a precision

$$\|\hat{y}\|_1 - \|\hat{x}\|_1 \leq e^{-N'/t}\varepsilon_0 \leq e^{-N/t}\varepsilon_0$$

and as $t \leq ht^*$

$$\|\hat{y}\|_1 - \|\hat{x}\|_1 \leq e^{-N/(ht^*)}\varepsilon_0$$

which gives the result. Finally the logarithmic grid search costs $\log_h N$ to get this approximative optimal bound. ∎

Sharpness therefore controls linear convergence of simple restart schemes to solve (ℓ_1 recovery). We now turn back to (NSP) estimates and connect them to recovery thresholds of the sampling matrix. This will give us a direct link between computational complexity and recovery performance for exact recovery problems.

4.1.3 Recovery threshold

If (NSP) is satisfied at a given order s it holds also for any $s' \leq s$. However, the constant, and therefore the speed of convergence, may change. Here we show that this constant actually depends on the ratio between the maximal order at which A satisfies (NSP) and the sparsity of the signal that we seek to recover.

To this end, we give a more concrete geometric meaning to the constant α in (NSP), connecting it with the diameter of a section of the ℓ_1 ball by the null space of the matrix A (see e.g. Kashin and Temlyakov [2007] for more details).

Lemma 4.1.6. *Given a matrix $A \in \mathbb{R}^{n\times p}$, denote*

$$\frac{1}{2}\,\mathbf{diam}(B_1^p \cap \mathrm{Null}(A)) = \sup_{\substack{Az=0 \\ \|z\|_1 \leq 1}} \|z\|_2,$$

the radius of the section of the ℓ_1 ball B_1^p by the null space of the matrix A and

$$s_A \triangleq 1/\,\mathbf{diam}(B_1^p \cap \mathrm{Null}(A))^2, \tag{4.8}$$

a recovery threshold. Then A satisfies (NSP) at any order $s < s_A$ with constant

$$\alpha = 2\sqrt{s_A/s} - 1 > 1. \tag{4.9}$$

Proof. For any $z \in \mathrm{Null}(A)$ and support set S with $\mathbf{Card}(S) \leq s$, using equivalence of norms and definition of the radius,

$$\|z_S\|_1 \leq \sqrt{s}\,\|z\|_2 \leq \frac{1}{2}\sqrt{\frac{s}{s_A}}\,\|z\|_1 = \frac{1}{2}\sqrt{\frac{s}{s_A}}\,(\|z_S\|_1 + \|z_{S^c}\|_1),$$

which means, as $s < s_A$,

$$\|z_{S^c}\|_1 \geq (2\sqrt{s_A/s} - 1)\|z_S\|_1,$$

hence the desired result. ∎

With s_A defined in (4.8), for any signal x^* of sparsity $s < s_A$, the sharpness bound (Sharp) then reads

$$\|x\|_1 - \|x^*\|_1 \geq \left(1 - \sqrt{s/s_A}\right) \|x - x^*\|_1,$$

and the optimal restart scheme defined in Proposition 4.1.4 has complexity

$$\|\hat{y}\|_1 - \|x^*\|_1 \leq \exp\left(-\left(1 - \sqrt{s/s_A}\right) \frac{e}{2\sqrt{p}} N\right) \varepsilon_0,$$

which means that, given a sensing matrix A with recovery threshold s_A, the sparser the signal, the faster the algorithm.

Precise estimates of the diameter of random sections of norm balls can be computed using classical results in geometric functional analysis. The low M^* estimates of Pajor and Tomczak-Jaegermann [1986] (see [Vershynin, 2011, Theorem 3.1] for a concise presentation) show that when $E \subset \mathbb{R}^p$ is a random subspace of codimension n (e.g. the null space of a random matrix $A \in \mathbb{R}^{n \times p}$), then

$$\mathbf{diam}(B_1^p \cap E) \leq c\sqrt{\frac{\log p}{n}},$$

with high probability, where $c > 0$ is an absolute constant. This means that the recovery threshold s_A satisfies

$$s_A \geq n/(c^2 \log p),$$

with high probability and leads to the following corollary.

Corollary 4.1.7. *Given a random sampling matrix $A \in \mathbb{R}^{n \times p}$ and a signal x^* with sparsity $s < n/(c^2 \log p)$, (Restart) scheme with optimal parameters defined in (4.4) outputs a point $\hat{y}$ such that*

$$\|\hat{y}\|_1 - \|x^*\|_1 \leq \exp\left(-\left(1 - c\sqrt{\frac{s \log p}{n}}\right) \frac{e}{2\sqrt{p}} N\right) \varepsilon_0,$$

with high probability, where c is a universal constant and N is the total number of iterations.

This means that the complexity of the optimization problem (ℓ_1 recovery) is controlled by the oversampling ratio n/s. In other words, while increasing the number of samples increases the time complexity of elementary operations of the algorithm, it also increases its rate of convergence.

4.2 A conic view for sparse recovery problems

We first gave concrete evidence of the link between optimization complexity and recovery performance for the exact recovery problem by highlighting sharpness properties of the objective around the true signal, given by the null space condition. We now take a step back and consider results on the underlying conic geometry of recovery problems that also control both computational and statistical aspects.

On the statistical side, minimal conically restricted singular values are known to control recovery performance in robust recovery problems. On the computational side, Renegar's condition number, a well known computational complexity measure for conic convex programs, controls the cost of obtaining optimality certificates for exact recovery and the sharpness of exact recovery problems (hence computational complexity of the (Restart) scheme presented in the previous section). Numerical experiments will then illustrate its relevance to control numerous other classical algorithms. By observing that minimal conically restricted singular values match the worst case of Renegar's condition number on sparse signals, our analysis shows once more that one single geometrical quantity controls both statistical robustness and computational complexity of recovery problems.

4.2.1 Conic linear systems

Conic linear systems arise naturally from optimality conditions of the exact recovery problem. To see this, define the tangent cone at point x with respect to the ℓ_1 norm, that is, the set of descent directions for $\|\cdot\|_1$ at x, as

$$\mathcal{T}(x) = \text{cone}\{z : \|x + z\|_1 \leq \|x\|_1\}. \tag{4.10}$$

As shown for example by Chandrasekaran et al. [2012, Prop 2.1] a point x is then the unique optimum of the exact recovery problem (ℓ_1 recovery) if and only if $\text{Null}(A) \cap \mathcal{T}(x) = \{0\}$, that is, there is no point satisfying the linear constraints that has lower ℓ_1 norm than x. Correct recovery of an original signal x^* is therefore certified by the infeasibility of a conic linear system of the form

$$\begin{array}{ll} \text{find} & z \\ \text{s.t.} & Az = 0 \\ & z \in C,\ z \neq 0, \end{array} \tag{$\text{P}_{\text{A,C}}$}$$

where C is a closed convex cone and A a given matrix. For both computational and statistical aspects we will be interested in the distance to feasibility. On the computational side this will give a distance to ill-posedness that plays the role of a condition number. On the statistical side it will measure the amount of perturbation that the recovery can handle.

Definition 4.2.1 (Distance to feasibility)**.** *Writing* $\mathcal{M}_C = \{A \in \mathbb{R}^{n\times p} : (\text{P}_{\text{A,C}})$ *is infeasible*$\}$,

distance to feasibility is defined as

$$\sigma_C(A) \triangleq \inf_{\Delta A} \{\|\Delta A\|_2 : A + \Delta A \notin \mathcal{M}_C\}. \tag{4.11}$$

A geometric analysis of the problem explicits the distance to feasibility in terms of minimal conically restricted singular value, as recalled in the following lemma.

Lemma 4.2.2. *Given a matrix $A \in \mathbb{R}^{p\times n}$ and a closed convex cone C, the distance to feasibility of* ($\mathrm{P}_{\mathrm{A,C}}$) *is given by*

$$\sigma_C(A) = \min_{\substack{x\in C \\ \|x\|_2=1}} \|Ax\|_2. \tag{4.12}$$

Proof. We recall the short proof of Amelunxen and Lotz [2014, Lemma 3.2]. Similar results have been derived by Freund and Vera [1999*b*, Theorem 2] and Belloni and Freund [2009, Lemma 3.2]. Let $z \in C$, with $\|z\|_2 = 1$, achieve the minimum above. Then $\Delta A = -Azz^T$ satisfies $(A + \Delta A)z = 0$, so $A + \Delta A \notin \mathcal{M}_C$ and

$$\sigma_C(A) \leq \|\Delta A\|_2 = \|Az\|_2\|z\|_2 = \min_{\substack{x\in C \\ \|x\|_2=1}} \|Ax\|_2.$$

On the other hand denote ΔA a perturbation such that $A + \Delta A \notin \mathcal{M}_C$. Then there exists $z \in C \setminus \{0\}$ such that $(A + \Delta A)z = 0$. Thus we have

$$\|\Delta A\|_2 \geq \frac{\|\Delta Az\|_2}{\|z\|_2} = \frac{\|Az\|_2}{\|z\|_2} \geq \min_{\substack{x\in C \\ \|x\|_2=1}} \|Ax\|_2.$$

Taking the infimum on the left-hand side over all ΔA such that $A + \Delta A \notin \mathcal{M}_C$ concludes the proof. ∎

Expression (4.12) writes distance to infeasibility as a cone restricted eigenvalue. Minimal cone restricted eigenvalues also directly characterize recovery performance as we recall now.

4.2.2 Recovery performance of robust recovery

The previous section showed that recovery of a signal x^* is ensured by infeasibility of the conic linear system ($\mathrm{P}_{\mathrm{A},\mathcal{T}(x^*)}$), i.e. positiveness of the minimal conically restricted singular value $\sigma_{\mathcal{T}(x^*)}(A)$. We now show how this quantity also controls recovery performance in the presence of noise. In that case, the robust recovery problem attempts to retrieve an original signal x^* by solving

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & \|Ax - b\|_2 \leq \delta\|A\|_2, \end{array} \qquad \text{(Robust } \ell_1 \text{ recovery)}$$

in the variable $x \in \mathbb{R}^p$, with the same design matrix $A \in \mathbb{R}^{n\times p}$, where $b \in \mathbb{R}^n$ are given observations perturbed by noise of level $\delta > 0$. The following classical result

then bounds reconstruction error in terms of $\sigma_{\mathcal{T}(x^*)}(A)$.

Lemma 4.2.3. *Given a coding matrix $A \in \mathbb{R}^{n\times p}$ and an original signal x^*, suppose we observe $b = Ax^* + w$ where $\|w\|_2 \leq \delta\|A\|_2$ and denote an optimal solution of (Robust ℓ_1 recovery) by $\hat{x}$. If the minimal singular value $\sigma_{\mathcal{T}(x^*)}(A)$ in (4.12) restricted to the tangent cone $\mathcal{T}(x^*)$ in (4.10) is positive, the following error bound holds:*

$$\|\hat{x} - x^*\|_2 \leq 2\frac{\delta\|A\|_2}{\sigma_{\mathcal{T}(x^*)}(A)}. \tag{4.13}$$

Proof. We recall the short proof of Chandrasekaran et al. [2012, Prop. 2.2]. Both $\hat{x}$ and x^* are feasible for (Robust ℓ_1 recovery) and $\hat{x}$ is optimal, so that $\|\hat{x}\|_1 \leq \|x^*\|_1$. Thus, the error vector $\hat{x} - x^*$ is in the tangent cone $\mathcal{T}(x^*)$. By the triangle inequality,

$$\|A(\hat{x} - x^*)\|_2 \leq \|A\hat{x} - b\|_2 + \|Ax^* - b\|_2 \leq 2\delta\|A\|_2.$$

Furthermore, by definition of $\sigma_{\mathcal{T}(x^*)}(A)$,

$$\|A(\hat{x} - x^*)\|_2 \geq \sigma_{\mathcal{T}(x^*)}(A)\,\|\hat{x} - x^*\|_2.$$

Combining the two concludes the proof. ∎

Therefore the robustness of the coding matrix A on all s-sparse signals is measured by

$$\mu_s(A) \triangleq \inf_{x\,:\,\|x\|_0\leq s} \min_{\substack{z\in\mathcal{T}(x)\\ \|z\|_2=1}} \|Az\|_2. \tag{4.14}$$

Expression of this minimal conically singular value can the be simplified by identifying the tangent cones on s-sparse signals, as done in the following lemma.

Lemma 4.2.4. *For any subset $S \subset [\![1, p]\!]$, let*

$$\mathcal{E}_S = \{z : \|z_{S^c}\|_1 \leq \|z_S\|\} \qquad \text{and} \qquad \mathcal{F}_S = \bigcup_{x\,:\,x=x_S} \mathcal{T}(x),$$

then $\mathcal{E}_S = \mathcal{F}_S$.

Proof. Let $z \in \mathcal{E}_S$, take $x = -z_S$, then

$$\|x + z\|_1 = \|z_{S^c}\|_1 \leq \|z_S\|_1 = \|x\|_1.$$

Therefore $z \in \mathcal{T}(x) \subset \mathcal{F}_S$ as $x = x_S$.

Conversely let $z \in \mathcal{F}_S$, and $x \in \mathbb{R}^p$, with $x = x_S$, such that $z \in \mathcal{T}(x)$. Then

$$\|x + z\|_1 = \|x + z_S\|_1 + \|z_{S^c}\|_1 \geq \|x\|_1 - \|z_S\|_1 + \|z_{S^c}\|_1.$$

As $z \in \mathcal{T}(x)$, this implies $\|z_{S^c}\|_1 \leq \|z_S\|_1$, so $z \in \mathcal{E}_S$ and we conclude that $\mathcal{E}_S = \mathcal{F}_S$.
∎

Therefore, the previous expression for the minimal conically restricted singular value (4.14) can be equivalently stated as

$$\mu_s(A) = \min_{\substack{S\subset[\![1,p]\!] \\ \mathbf{Card}(S)\leq s}} \min_{\substack{\|z_{S^c}\|_1\leq\|z_S\|_1 \\ \|z\|_2=1}} \|Az\|_2. \tag{4.15}$$

This quantity was introduced in [Bickel et al., 2009] and further explored in e.g. [Van De Geer et al., 2009]. Bickel et al. [2009] notably showed that it controls estimation performance of LASSO and Dantzig selector. Observe that positiveness of $\mu_s(A)$ is equivalent to (NSP) at order s with constant 1 which shows necessity of (NSP) for sparse recovery.

Since both null space property and conically restricted singular values are necessary and sufficient conditions for exact recovery they may have been linked previously in the literature. Here we derive estimates for the constant in (NSP) from the minimal cone restricted singular value using tools form conic linear systems. We search for α such that (NSP) is satisfied at order s. Equivalently we search for α such that for any support S of cardinality at most s, the conic linear system

$$\begin{array}{ll} \text{find} & z \\ \text{s.t.} & Az = 0 \\ & \|z_{S^c}\|_1 \leq \alpha\|z_S\|_1,\ z \neq 0 \end{array} \tag{4.16}$$

is infeasible. Notice that system (4.16) for $\alpha > 1$ is a perturbed version of the case $\alpha = 1$, so the problem reduces to studying the sensitivity to perturbations of conic linear systems as shown in the following lemma.

Lemma 4.2.5. *Given a matrix $A \in \mathbb{R}^{n\times p}$ and an integer $s \in [\![1,p]\!]$, if the minimal conically restricted singular value $\mu_s(A)$ in (4.14) and (4.15) is positive, then A satisfies (NSP) at order s for any constant*

$$\alpha \leq \left(1 - \frac{\mu_s(A)}{\|A\|_2}\right)^{-1}.$$

Proof. For a support S of cardinality at most s, write P the orthogonal projector on this support (that is, $Px = x_S$), $\bar{P} = \mathbf{I} - P$ its orthogonal projector and define the closed convex cone $C_S = \{z : \|z_{S^c}\|_1 \leq \|z_S\|_1\}$. Given $\alpha \geq 1$, denote $H = \alpha^{-1}P + \bar{P} = \mathbf{I} - (1-\alpha^{-1})P$. Observe that

$$\{z : \|z_{S^c}\|_1 \leq \alpha\|z_S\|_1\} = HC_S.$$

Therefore, the conic linear system (4.16) reads

$$\begin{array}{ll} \text{find} & z \\ \text{s.t.} & Az = 0 \\ & z \in HC_S,\ z \neq 0. \end{array}$$

As H is invertible, this is equivalent to

$$\begin{array}{ll} \text{find} & z \\ \text{s.t.} & AHz = 0 \\ & z \in C_S,\ z \neq 0. \end{array} \tag{4.17}$$

Therefore, if the conic linear system

$$\begin{array}{ll} \text{find} & z \\ \text{s.t.} & Az = 0 \\ & z \in C_S,\ z \neq 0 \end{array}$$

is infeasible, that is $\sigma_{C_S}(A) > 0$, by Lemma 4.2.2, which is true for $\mu_s > 0$, then by definition of the distance to feasibility, (4.17) is also infeasible provided $\|AH - A\|_2 \leq \sigma_{C_S}(A)$, which holds for any $\alpha \geq 1$ such that

$$(1 - \alpha^{-1})\|AP\|_2 \leq \sigma_{C_S}(A).$$

Using that $\|AP\|_2 \leq \|A\|_2$, infeasibility is ensured in particular for any α such that

$$1 - \frac{\sigma_{C_S}(A)}{\|A\|_2} \leq \alpha^{-1}.$$

To ensure infeasibility of the conic linear systems (4.16) for any support S, it suffices to take α such that

$$1 - \frac{\mu_s(A)}{\|A\|_2} \leq \alpha^{-1}.$$

This means that (NSP) at order s is satisfied for any

$$\alpha \leq \left(1 - \frac{\mu_s(A)}{\|A\|_2}\right)^{-1}$$

where we used that, by definition of the minimal conically restricted singular value, $\mu_s(A) \leq \|A\|_2$ (in case of equality (NSP), will be satisfied for any $\alpha \geq 1$). ∎

We now relate the minimal cone restricted singular value to computational complexity measures.

4.2.3 Computational complexity of recovery problems

Computational complexity for convex optimization problems is often described in terms of polynomial functions of problem size. This produces a clear link between problem structure and computational complexity but fails to account for the nature of the data. If we use linear systems as a basic example, unstructured linear systems of dimension n can be solved with complexity $O(n^3)$ regardless of the matrix values, but iterative solvers will converge much faster on systems that are better conditioned. The seminal work of Renegar [1995*b*, 2001] extends this notion of conditioning to

optimization problems, producing data-driven bounds on the complexity of solving conic programs, and showing that the number of outer iterations of interior point algorithms increases as the distance to ill-posedness decreases.

Renegar's condition number

Renegar's condition number [Renegar, 1995*b*,*a*; Peña, 2000] provides a data-driven measure of the complexity of certifying infeasibility of a conic linear system of the form presented in ($\mathrm{P_{A,C}}$) (the larger the condition number, the harder the problem). It is rooted in the sensible idea that certifying infeasibility is easier if the problem is far from being feasible. It is defined as the scale invariant reciprocal of the distance to feasibility $\sigma_C(A)$, defined in (4.11), of problem ($\mathrm{P_{A,C}}$), i.e.

$$\mathcal{R}_C(A) \triangleq \frac{\|A\|_2}{\sigma_C(A)} = \|A\|_2 / \min_{\substack{x \in C \\ \|x\|_2 = 1}} \|Ax\|_2. \tag{4.18}$$

Notice that, if C were the whole space $\mathbb{R}^p$, and if A^TA were full-rank (never the case if $n < p$), then $\sigma_C(A)$ would be the smallest singular value of A. As a result, $\mathcal{R}_C(A)$ would reduce to the classical condition number of A (and to ∞ when A^TA is rank-deficient). Renegar's condition number is necessarily smaller (better) than the latter, as it further incorporates the notion that A need only be well conditioned along those directions that matter with respect to C.

Complexity of certifying optimality

In a first step, we study the complexity of the oracle certifying optimality of a candidate solution x to (ℓ_1 recovery) as a proxy for the problem of computing an optimal solution to this problem. As mentioned in Section 4.2.1, optimality of a point x is equivalent to infeasibility of

$$\begin{array}{ll} \text{find} & z \\ \text{s.t.} & Az = 0 \\ & z \in \mathcal{T}(x),\ z \neq 0, \end{array} \tag{$\mathrm{P_{A,\mathcal{T}(x)}}$}$$

where the tangent cone $\mathcal{T}(x)$ is defined in (4.10). By a theorem of alternative, infeasibility of ($\mathrm{P_{A,\mathcal{T}(x)}}$) is equivalent to feasibility of the dual problem

$$\begin{array}{ll} \text{find} & y \\ \text{s.t.} & A^Ty \in \mathbf{int}(\mathcal{T}(x)^\circ), \end{array} \tag{$\mathrm{D_{A,\mathcal{T}(x)}}$}$$

where $\mathcal{T}(x)^\circ$ is the polar cone of $\mathcal{T}(x)$. Therefore, to certify infeasibility of ($\mathrm{P_{A,\mathcal{T}(x)}}$) it is sufficient to exhibit a solution for the dual problem ($\mathrm{D_{A,\mathcal{T}(x)}}$).

Several references have connected Renegar's condition number and the complexity of solving such conic linear systems using various algorithms [Renegar, 1995*b*; Freund and Vera, 1999*a*; Epelman and Freund, 2000; Renegar, 2001; Vera et al., 2007; Belloni et al., 2009]. In particular, Vera et al. [2007] linked it to the complexity of solving

the primal dual pair ($P_{A,\mathcal{T}(x)}$)–($D_{A,\mathcal{T}(x)}$) using a barrier method. They show that the number of outer barrier method iterations grows as

$$O\left(\sqrt{\rho}\log\left(\rho\,\mathcal{R}_{\mathcal{T}(x)}(A)\right)\right),$$

where ρ is the barrier parameter, while the conditioning (hence the complexity) of the linear systems arising at each interior point iteration is controlled by $\mathcal{R}_{\mathcal{T}(x)}(A)^2$. This link was also tested empirically on linear programs using the NETLIB library of problems by Ordóñez and Freund [2003], where computing times and number of iterations were regressed against estimates of the condition number computed using the approximations for Renegar's condition number detailed by Freund and Vera [2003].

Studying the complexity of computing an optimality certificate gives insights on the performance of oracle based optimization techniques such as the ellipsoid method. We now show how Renegar's condition also controls the number steps in the (Restart) scheme presented in Section 4.1.2.

Complexity of restart scheme with Renegar's condition number

Convergence of the (Restart) scheme presented in Section 4.1.2 is controlled by the sharpness of the problem deduced from (NSP). We now observe that sharpness is controlled by the worst case Renegar condition number for the optimality certificates ($P_{A,\mathcal{T}(x)}$) on all s-sparse signals, defined as

$$\mathcal{R}_s(A) \triangleq \sup_{x\,:\,\|x\|_0\leq s} \mathcal{R}_{\mathcal{T}(x)}(A) = \|A\|_2/\mu_s(A). \tag{4.19}$$

Connecting Lemmas 4.2.3, 4.2.5 and Proposition 4.1.4 we get the following corollary.

Corollary 4.2.6. *Given a coding matrix $A \in \mathbb{R}^{n\times p}$ and a sparsity level $s \geq 1$, if $\mathcal{R}_s(A) < +\infty$ in (4.19) then optimal (Restart) scheme achieves an ε precision in at most*

$$O((2\mathcal{R}_s(A) - 1)\log\varepsilon^{-1})$$

iterations.

Proof. From Lemma 4.2.3, if $\mathcal{R}_s(A) < +\infty$, then (NSP) is satisfied for $\alpha = (1 - \mathcal{R}_s(A)^{-1})^{-1} > 1$, so from Lemma 4.2.5 (4.1) is sharp with constant $\gamma = \frac{\alpha-1}{\alpha+1} = (2\mathcal{R}_s(A) - 1)^{-1}$ and Proposition 4.1.4 leads to the linear convergence rate above. ∎

This shows that Renegar's condition number explicitly controls the convergence of an algorithmic scheme devoted to the exact recovery problem (ℓ_1 recovery), through its link with sharpness.

On the statistical side, we observed that the minimal conically restricted singular value controls recovery performance of robust procedures and that its positivity ensures exact recovery. On the computational side, we presented the role of Renegar's condition number as a computational complexity measure for sparse recovery problems. A key observation is that the worst case of Renegar's condition number $\mathcal{R}_s(A)$,

defined in (4.19), matches the minimal conically restricted singular value defined in (4.14). Once again, a single quantity controls both aspects. This at least partially explains the common empirical observation (see, e.g., Donoho and Tsaig [2008]) that problem instances where statistical estimation succeeds are computationally easier to solve.

4.2.4 Computational complexity for inexact recovery

When the primal problem ($P_{A,\mathcal{T}(x)}$) is feasible, so that $\sigma_{\mathcal{T}(x)}(A) = 0$, Renegar's condition number as defined here is infinite. While this correctly captures the fact that, in that regime, statistical recovery does not hold, it does not properly capture the fact that, when ($P_{A,\mathcal{T}(x)}$) is "comfortably" feasible, certifying so is easy, and algorithms terminate quickly (although they return a useless estimator). From both a statistical and a computational point of view, the truly delicate cases correspond to problem instances for which both ($P_{A,\mathcal{T}(x)}$) and ($D_{A,\mathcal{T}(x)}$) are only barely feasible or infeasible. This is illustrated in simple numerical example by Boyd and Vandenberghe [2004, §11.4.3] and in our numerical experiments, corresponding to the peaks in the CPU time plots of the right column in Figure 4-4: problems where sparse recovery barely holds/fails are relatively harder. For simplicity, we only focused here on distance to feasibility for problem ($P_{A,\mathcal{T}(x)}$). However, it is possible to symmetrize the condition numbers used here as described by Amelunxen and Lotz [2014, §1.3], where a symmetric version of the condition number is defined as

$$\bar{\mathcal{R}}_{\mathcal{T}(x)}(A) = \min\left\{\frac{\|A\|}{\sigma^P_{\mathcal{T}(x)}(A)}, \frac{\|A\|}{\sigma^D_{\mathcal{T}(x)}(A)}\right\},$$

where $\sigma^P_{\mathcal{T}(x)}(A)$ and $\sigma^D_{\mathcal{T}(x)}(A)$ denote the distance to feasibility of respectively ($P_{A,\mathcal{T}(x)}$) and ($D_{A,\mathcal{T}(x)}$). This quantity peaks for programs that are nearly feasible/infeasible.

As we noticed in Section 4.1.2, a Łojasiewicz inequality (4.6) for the (ℓ_1 recovery) problem is sufficient to ensure linear convergence of the restart scheme. Connecting the symmetrized Renegar condition number to the Łojasiewicz inequality constant γ may then produce complexity bounds for the restart scheme beyond the recovery case. Łojasiewicz inequalities for convex programs have indeed proven their relevance. They were used by Fercoq and Qu [2016]; Roulet and d'Aspremont [2017] to accelerate classical methods, in particular on the LASSO problem. Lower computational bounds for convex optimization problems satisfying it were also studied by Nemirovskii and Nesterov [1985, Page 6]. Although the Łojasiewicz inequality is proven to be satisfied by a broad class of functions [Bolte et al., 2007], quantifying its parameters is still a challenging problem that would enable better parameter choices for appropriate algorithms.

4.2.5 Other algorithms

The restart scheme presented in Section 4.1.2 is of course not the only one to solve problem (ℓ_1 recovery) in practice and it has not been analyzed in the noisy

case. However, we will observe in the numerical experiments of Section 4.4 that the condition number is correlated with the empirical performance of efficient recovery algorithms such as LARS [Efron et al., 2004] and Homotopy [Donoho and Tsaig, 2008; Asif and Romberg, 2014]. On paper, the computational complexities of (ℓ_1 recovery) and (Robust ℓ_1 recovery) are very similar (in fact, infeasible start primal-dual algorithms designed for solving (ℓ_1 recovery) actually solve problem (Robust ℓ_1 recovery) with δ small). However in our experiments, we did observe sometimes significant differences in behavior between the noisy and noiseless case.

4.3 Generalization to common sparsity inducing norms

In this section we generalize previous results to sparse recovery problems in (non-overlapping) group norms or nuclear norm. Group norms arise in contexts such as genomics to enforce the selection of groups of genes (e.g., Obozinski et al. [2011] and references therein.) The nuclear norm is used for low-rank estimation (e.g., Recht et al. [2008] and references therein.) We use the framework of decomposable norms introduced by Negahban et al. [2009] which applies to these norms. This allows us to generalize the null space property and to derive corresponding sharpness bounds for the exact recovery problem in a broader framework. We then again relate recovery performance and computational complexity of these recovery problems.

4.3.1 Decomposable norms

Sparsity inducing norms have been explored from various perspectives. Here, we use the framework of decomposable norms by Negahban et al. [2009] to generalize our results from ℓ_1 norms to non-overlapping group norms and nuclear norms in a concise form. We then discuss the key geometrical properties of these norms and potential characterization of their conic nature.

We first recall the definition of decomposable norms by Negahban et al. [2009] in terms of projectors.

Definition 4.3.1. Decomposable norms *Given a Euclidean space E, a norm $\|.\|$ on E is said to be decomposable if there exists a family of orthogonal projectors $\mathcal{P}$ such that*

(i) to each $P \in \mathcal{P}$ is associated a non-negative weight $\eta(P)$ and an orthogonal projector $\bar{P}$ such that $P\bar{P} = \bar{P}P = 0$, and

(ii) for any $x \in E$ and $P \in \mathcal{P}$, $\|Px + \bar{P}x\| = \|Px\| + \|\bar{P}x\|$.

A signal x is then said to be s-sparse if there exists $P \in \mathcal{P}$, such that $\eta(P) \leq s$ and $Px = x$.

We now detail the family of projectors for some decomposable norms of interest.

ℓ_1 norm.

In the the ℓ_1 norm case, $E = \mathbb{R}^p$ and $\mathcal{P}$ is the set of projectors on coordinate subspaces of $\mathbb{R}^p$, that is, $\mathcal{P}$ contains all projectors which zero out all coordinates of a vector except for a subset of them, which are left unaffected. The maps $\bar{P}$ are the complementary projectors: $\bar{P} = \mathbf{I} - P$. Property (ii) is the classical decomposability of the ℓ_1 norm. Naturally, the complexity level corresponds to the number of coordinates preserved by P, i.e., $\nu(P) = \mathbf{Rank}(P)$. These definitions recover the usual notion of sparsity.

Group norms.

Given a partition G of $[\![1, p]\!]$ in (non-overlapping) groups $g \subset [\![1, p]\!]$, the group norm is defined for $x \in \mathbb{R}^p$ as

$$\|x\| = \sum_{g \in G} \|x_g\|_r,$$

where $\|x_g\|_r$ is the ℓ_r-norm of the projection of x onto the coordinates defined by g. The cases $r = 2, \infty$ correspond respectively to ℓ_1/ℓ_2 and ℓ_1/ℓ_∞ block norms. Here, $E = \mathbb{R}^p$ and the family $\mathcal{P}$ is composed of orthogonal projectors onto coordinates defined by (disjoint) unions of groups g, and $\bar{P} = \mathbf{I} - P$. Formally, to each P we associate $F \subset G$ such that for any $x \in E$, $(Px)_g = x_g$ if $g \in F$ and $(Px)_g = 0$ otherwise. Decomposability (ii) then clearly holds. To each group g we associate a weight η_g and for a projector $P \in \mathcal{P}$ with associated $F \subset G$, $\eta(P) = \sum_{g \in F} \eta_g$. A classical choice of weights is $\eta_g = 1$ for all $g \in G$.

Nuclear norm.

The nuclear norm is defined for matrices $X \in \mathbb{R}^{p \times q}$ with singular values $\sigma_i(X)$ as

$$\|X\| = \sum_{k=1}^{\min(p,q)} \sigma_k(X).$$

Here $E = \mathbb{R}^{p \times q}$ and its associated family of projectors contains P such that

$$P \colon X \mapsto P_{\text{left}} X P_{\text{right}},$$

and

$$\bar{P} \colon X \mapsto (\mathbf{I} - P_{\text{left}}) X (\mathbf{I} - P_{\text{right}}),$$

where $P_{\text{left}} \in \mathbb{R}^{p \times p}$ and $P_{\text{right}} \in \mathbb{R}^{q \times q}$ are orthogonal projectors. Their weights are defined as $\eta(P) = \max\left(\mathbf{Rank}(P_{\text{left}}), \mathbf{Rank}(P_{\text{right}})\right)$ defining therefore s-sparse matrices as matrices of rank at most s. As P and $\bar{P}$ project on orthogonal row and column spaces, condition (ii) holds.

Decomposable norms offer a unified nomenclature for the study of sparsity inducing norms. However, they appear to be essentially restricted to the three examples presented above. Moreover, it is not clear if their definition is sufficient to character-

ize the conic nature of these norms, in particular in the nuclear norm case that will require additional linear algebra results. In comparison, the framework proposed by Juditsky et al. [2014] can encompass *non-latent* overlapping groups. For future use, we simplify the third property of their definition [Juditsky et al., 2014, Section 2.1] in Appendix 4.B. It is not clear how this view can be used for latent overlapping group norms presented by Obozinski et al. [2011] applied in biology. Moreover the sufficient conditions that Juditsky et al. [2014] present are sufficient but not necessary in the nuclear norm case. Better characterizing the key geometrical properties of these norms is therefore a challenging research direction.

4.3.2 Sharpness and generalized null space property

From now on, we assume that we are given an ambient Euclidean space E with one of the three decomposable norms $\|.\|$ presented in previous section, i.e. ℓ_1, group or nuclear norm, and the associated family of orthogonal projectors $\mathcal{P}$ as introduced in Definition 4.3.1. We study the sparse recovery problem

$$\begin{array}{ll} \text{minimize} & \|x\| \\ \text{subject to} & A(x) = b \end{array} \tag{Sparse recovery}$$

in the variable $x \in E$, where A is a linear operator onto $\mathbb{R}^n$ and the observations $b \in \mathbb{R}^n$ are taken from an original point x^* such that $b = A(x^*)$. We begin by generalizing the null space property in this setting.

Definition 4.3.2. (Generalized Null space Property) *A linear operator A on E satisfies the Generalized Null Space Property (GNSP)* for orthogonal projector $P \in \mathcal{P}$ *with constant $\alpha \geq 1$ if and only if for any $z \in \mathrm{Null}(A) \setminus \{0\}$ such that $z = Pz + \bar{P}z$,*

$$\alpha \|Pz\| < \|\bar{P}z\|. \tag{GNSP}$$

The linear operator A satisfies the Generalized Null Space Property at order s *with constant $\alpha \geq 1$ if it satisfies it for any P such that $\eta(P) \leq s$.*

Notice that if $\bar{P} = \mathbf{I} - P$, the condition $z = Pz + \bar{P}z$ is not restrictive. However it will be useful to prove necessity of (GNSP) for the nuclear norm. In that case, observe that it is equivalent to the condition introduced by Oymak and Hassibi [2010], i.e.

$$\forall z \in \mathrm{Null}(A) \setminus \{0\}, \quad \alpha \sum_{i=1}^{s} \sigma_i(z) < \sum_{i=s+1}^{\min(p,q)} \sigma_i(z),$$

where $\sigma_i(z)$ are the singular values of z in decreasing order. Notice also that we recover the classical Definition NSP in the ℓ_1 case. The sharpness bound then easily follows if $\bar{P} = \mathbf{I} - P$. In the case of the nuclear norm it requires additional linear algebra results.

Proposition 4.3.3. *Given a linear operator A that satisfies* (GNSP) *at order s with constant α, if the original point x^* is s-sparse, then for any $x \in E$ satisfying $A(x) = b$,*

$x \neq x^*$, we have

$$\|x\| - \|x^*\| > \frac{\alpha - 1}{\alpha + 1}\|x - x^*\|.$$

This implies recovery, i.e., optimality of x^* for (Sparse recovery).

Proof. Denote P such that $\eta(P) \leq s$ and $Px^* = x^*$, which defines its sparsity. Let $x \neq x^*$ such that $A(x) = b$, so $z = x - x^* \in \text{Null}(A)$ and $z \neq 0$. If $\bar{P} = \mathbf{I} - P$, $x = Px + \bar{P}x$ and using the decomposability (ii), we have

$$\begin{aligned}\|x\| &= \|Px^* + Pz\| + \|\bar{P}z\| \\ &\geq \|x^*\| - \|Pz\| + \|\bar{P}z\| \\ &= \|x^*\| + \|z\| - 2\|Pz\|.\end{aligned}$$

By using (GNSP), $\|z\| = \|Pz\| + \|\bar{P}z\| > (1+\alpha)\|Pz\|$. The result follows by arranging the terms.

If $\|.\|$ is the nuclear norm and $\bar{P} \neq \mathbf{I} - P$, as in [Oymak and Hassibi, 2010, Lemma 6], we use that (see Horn and Johnson [1990, Theorem 7.4.9.1])

$$\|x^* + z\| \geq \sum_{i=1}^{\min(p,q)} |\sigma_i(x^*) - \sigma_i(z)|,$$

where $\sigma_i(x^*), \sigma_i(z)$ denote the singular values in decreasing order of respectively x^* and z. Then, using that x^* has rank at most s,

$$\begin{aligned}\|x\| &\geq \sum_{i=1}^{s} |\sigma_i(x^*) - \sigma_i(z)| + \sum_{i=s+1}^{\min(p,q)} \sigma_i(z) \\ &\geq \sum_{i=1}^{s} \sigma_i(x^*) - \sum_{i=1}^{s} \sigma_i(z) + \sum_{i=s+1}^{\min(p,q)} \sigma_i(z) \\ &= \|x^*\| - \|Qz\| + \|\bar{Q}z\|,\end{aligned}$$

where Q is the projector on the s largest singular directions of z and therefore $\bar{Q}$ the projector on the $n - s$ others. These can be defined using the singular value decomposition of z such that $z = Qz + \bar{Q}z$. Then, using (GNSP) and the decomposability (ii) concludes the proof as above. ∎

This shows that the sharpness bound of the form (Sharp) generalizes to non-overlapping group norms and the nuclear norm. Proposition 4.1.3 can also be generalized directly to this case with our definition of (GNSP). The smoothing argument and restart schemes developed in Section 4.1.2 can then be applied with similar linear convergence rates that essentially depend on the sharpness constant. By looking at the diameter of the section of the unit ball of the norm by the null space of A, one may also show that the oversampling ratio controls the sharpness bound as in Section 4.1.3.

As in Section 4.2, we now study the conic quantities which control statistical and optimization aspects.

4.3.3 Robust recovery performance and computational complexity

In this section, for a Euclidean space E and $x \in E$ we denote $\|x\|_2$ the ℓ_2 norm of its coefficients, if E is a matrix space $\|x\|_2$ is then the Frobenius norm of x.

Generalized cone restricted singular value

We begin by addressing the recovery performance of robust sparse recovery problems that reads

$$\begin{array}{ll} \text{minimize} & \|x\| \\ \text{subject to} & \|A(x) - b\|_2 \leq \delta \|A\|_2, \end{array} \qquad \text{(Robust sparse recovery)}$$

in the variable $x \in E$, with the same linear operator A, where the observations $b \in \mathbb{R}^n$ are affected by noise of level $\delta > 0$. For a linear operator A from E to $\mathbb{R}^n$, we denote its operator norm with respect to $\|\cdot\|$, $\|A\|_2 = \sup_{x \in E : \|x\|_2 \leq 1} \|A(x)\|_2$.

The results of Section 4.2.2 transpose directly to the general case by replacing $\|\cdot\|_1$ by $\|\cdot\|$. Precisely, assuming that $b = Ax^* + w$ where $\|w\|_2 \leq \delta \|A\|_2$, an optimal solution $\hat{x}$ of problem (Robust sparse recovery) satisfies the error bound

$$\|\hat{x} - x^*\|_2 \leq 2 \frac{\delta \|A\|_2}{\sigma_{\mathcal{T}(x^*)}(A)},$$

where the tangent cone is defined as

$$\mathcal{T}(x) = \operatorname{cone}\{z : \|x + z\| \leq \|x\|\},$$

and robust recovery of s-sparse signals is therefore controlled by

$$\mu_s(A) = \inf_{P \in \mathcal{P} : \eta(P) \leq s} \inf_{x \in E : Px = x} \min_{\substack{z \in \mathcal{T}(x) \\ \|z\|_2 = 1}} \|Az\|_2. \tag{4.20}$$

The key point is then to characterize the tangent cones of s-sparse signals. First, this will allow statistical estimations of $\mu_s(A)$. Second, it will enable us to estimate the constant (GNSP), hence sharpness of the exact recovery problem and computational complexity of associated restart schemes. This is the aim of the following lemma.

Lemma 4.3.4. *For a given sparsity s, write*

$$\mathcal{E} = \bigcup_{P \in \mathcal{P} : \eta(P) \leq s} \{z \in E \,:\, z = Pz + \bar{P}z,\ \|\bar{P}z\| \leq \|Pz\|\}$$

and

$$\mathcal{F} = \bigcup_{P\in\mathcal{P}\,:\,\eta(P)\leq s} \;\bigcup_{x\in E\,:\,x=Px} \mathcal{T}(x).$$

Then $\mathcal{E} = \mathcal{F}$.

Proof. Let $z \in \mathcal{E}$ and $P \in \mathcal{P}$ such that $z = Pz + \bar{P}z$. Taking $x = -Pz$ we get

$$\|x + z\| = \|\bar{P}z\| \leq \|Pz\| = \|x\|.$$

Therefore $z \in \mathcal{T}(x) \subset \mathcal{F}$. Conversely, if $z \in \mathcal{F}$, denote $x \in E$ and $P \in \mathcal{P}$ such that $x = Px$, $z \in \mathcal{T}(x)$ and $\eta(P) \leq s$. If $\bar{P} = \mathbf{I} - P$, by decomposability (ii),

$$\|x + z\| = \|Px + Pz\| + \|\bar{P}z\| \geq \|x\| - \|Pz\| + \|\bar{P}z\|.$$

Since $z \in \mathcal{T}(x)$, we have $\|x + z\| \leq \|x\|$; combined with the previous statement, this implies that $z \in \{z \in E : z = Pz + \bar{P}z,\ \|\bar{P}z\| \leq \|Pz\|\} \subset \mathcal{E}$. Now, if $\|.\|$ is the nuclear norm, as in the proof of Proposition (4.3.3), we have

$$\|x + z\| \geq \|x\| - \|Qz\| + \|\bar{Q}z\|,$$

where Q is the projector on the s largest singular directions of z given by the singular value decomposition of z, so that $z = Qz + \bar{Q}z$. Therefore, $z \in \mathcal{T}(x)$ implies $z \in \{z \in E : z = Qz + \bar{Q}z,\ \|\bar{Q}z\| \leq \|Qz\|\} \subset \mathcal{E}$. In all cases we have therefore proven $\mathcal{E} = \mathcal{F}$. ■

Using the previous lemma, the minimal cone restricted singular value reads:

$$\mu_s(A) = \inf_{P\in\mathcal{P},\,\eta(P)\leq s} \;\min_{\substack{z\in E,\,\|z\|_2=1 \\ z=Pz+\bar{P}z,\,\|\bar{P}z\|\leq\|Pz\|}} \|Az\|_2. \tag{4.21}$$

This quantity can then be linked to the (GNSP) constant, as shown in the following lemma.

Lemma 4.3.5. *Given a linear operator A on E, If the minimal cone restricted singular value $\mu_s(A)$, defined in (4.20) and reformulated in (4.21), is positive, then A satisfies (GNSP) at order s for any constant*

$$\alpha \leq \left(1 - \frac{\mu_s(A)}{\|A\|_2}\right)^{-1}.$$

Proof. For a given $P \in \mathcal{P}$, denote $C_P = \{z \in \Im(P) + \Im(\bar{P}) \;:\; \|\bar{P}z\| \leq \|Pz\|\}$ and define for $\alpha \geq 1$ the conic linear system

$$\begin{array}{ll} \text{find} & z \in \Im(P) + \Im(\bar{P}) \\ \text{s.t.} & A(z) = 0 \\ & \|\bar{P}z\| \leq \alpha\|Pz\|,\ z \neq 0. \end{array} \tag{4.22}$$

Infeasibility of this system for all $P \in \mathcal{P}$ such that $\eta(P) \leq s$ is then equivalent to (GNSP) at order s with constant α. Denote $H = \mathbf{I} - (1 - \alpha^{-1})P$ such that

$$\{z \in \mathfrak{S}(P) + \mathfrak{S}(\bar{P}) : \|\bar{P}z\| \leq \alpha \|Pz\|\} = HC_P.$$

Since H is invertible, we observe as in Lemma 4.2.5 that the conic linear system (4.22) is equivalent to

$$\begin{array}{ll} \text{find} & z \in \mathfrak{S}(P) + \mathfrak{S}(\bar{P}) \\ \text{s.t.} & A - (1 - \alpha^{-1})APz = 0 \\ & z \in C_P, \ z \neq 0. \end{array} \tag{4.23}$$

If this problem is infeasible for $\alpha = 1$, i.e., its distance to feasibility $\mu_{C_P}(A)$ defined in (4.11) is positive, then (4.23) is infeasible for any $\alpha \geq 1$ such that

$$(1 - \alpha^{-1})\|AP\| \leq \mu_{C_P}(A).$$

Now, if $\mu_s(A) > 0$ the conic linear system (4.22) will still be infeasible for any

$$\alpha \leq \left(1 - \frac{\mu_s(A)}{\|A\|_2}\right)^{-1}.$$

Thus, A satisfies (GNSP) at order s with α as above. ∎

Renegar's condition number

On the computational side, denote $\mathcal{R}_{\mathcal{T}(x)}(A)$ the Renegar condition number of the conic linear system

$$\begin{array}{ll} \text{find} & z \\ \text{s.t.} & A(z) = 0 \\ & z \in \mathcal{T}(x), \ z \neq 0, \end{array}$$

and the worst-case Renegar condition number on s-sparse signals

$$\mathcal{R}_s(A) \triangleq \sup_{P \in \mathcal{P} \,:\, \eta(P) \leq s} \ \sup_{x \in E \,:\, Px = x} \mathcal{R}_{\mathcal{T}(x)}(A) = \|A\|_2 / \mu_s(A).$$

First, Renegar's condition number plays the same role as before in computing optimality certificates for the exact recovery problems. Then, combining Lemma 4.3.5 and Proposition 4.3.3 shows that the sharpness bound for exact recovery reads

$$\|x\| - \|x^*\| > \frac{1}{2\mathcal{R}_s(A) - 1}\|x - x^*\|.$$

This sharpness will then control linearly convergent restart schemes for the exact recovery problem.

Overall then, as established earlier in this paper, a single geometric quantity—namely, the minimal cone restricted singular value—appears to control both computational and statistical aspects. We now illustrate this statement on numerical

experiments.

4.4 Numerical results

In this section, we first test the empirical performance of restart schemes and its link with recovery performance. We then perform similar experiments on Renegar's condition number.

4.4.1 Sharpness & restart for exact recovery

We test the (Restart) scheme on ℓ_1-recovery problems with random design matrices. Throughout the experiments, we use the NESTA code described in [Becker, Bobin and Candès, 2011] as the subroutine in the restart strategy. We generate a random design matrix $A \in \mathbb{R}^{n \times p}$ with i.i.d. Gaussian coefficients. We then normalize A so that $AA^T = \mathbf{I}$ (to fit NESTA's format) and generate observations $b = Ax^*$ where $x^* \in \mathbb{R}^p$ is an s-sparse vector whose nonzero coefficients are all ones. We denote $\hat{x}$ the solution given by a common solver run at machine precision and plot convergence $f(x_t) - f^* = \|x_t\|_1 - \|\hat{x}\|_1$ (scaled such that $f(x_0) - f(\hat{x}) = 1$).

Restart scheme performance

First we compare in Figure 4-1 the practical scheme presented in Section 4.1.2 with a plain implementation of NESTA without restart or continuation steps. Dimensions of the problem are $p = 500$, $n = 200$ and $s = 30$. Starting from $x_0 = A^T b$, we use $\varepsilon_0 = \|x_0\|_1$ as a first initial guess on the gap and perform a grid search of step size $h = 4$ for a budget of $N = 500$ iterations. The first and last schemes of the grid search were not run as they are unlikely to produce a nearly optimal restart scheme. The grid search can be parallelized and the best scheme found is plotted with a solid red line. The dashed red line represents the convergence rate accounting for the cost of the grid search. For the plain implementation of NESTA, we used different target precisions. These control indeed the smoothness of the surrogate function f_ε which itself controls the step size of Nesterov's algorithm. Therefore a high precision slows down the algorithm. However for low precision NESTA can be faster but will not approximate well the original signal. Also, the theoretical bound (4.2) might be very pessimistic, as the surrogate function f_ε may approximate the ℓ_1 norm for the points of interest at a much better accuracy than ε.

Overall, we observe a clear linear convergence of the restart scheme that outperforms the plain implementation. This was already observed by Becker, Bobin and Candès [2011] who developed their continuation steps against which we compare in Figure 4-2. We used default options for NESTA, namely 5 continuation steps with a stopping criterion based on the relative objective change in the surrogate function (specifically, the algorithm stops when these changes are lower than the target accuracy, set to 10^{-6}). We compare continuations steps and best restart found by grid search for different dimensions of the problem, we fix $p = 500$, $s = 30$ and vary the

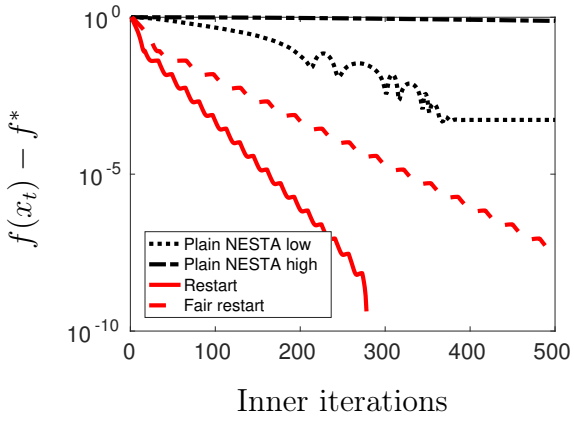

Figure 4-1 – Best restarted NESTA (solid red line) and overall cost of the practical restart schemes (dashed red line) versus plain NESTA implementation with low accuracy $\varepsilon = 10^{-1}$ (dotted black line) and higher accuracy $\varepsilon = 10^{-3}$ (dash-dotted black line) for a budget of 500 iterations.

number of samples $n = \{200, 300\}$. Continuation steps converge faster with better conditioned problems, i.e., more samples. Otherwise they may get stuck due to the termination criterion. Notice that a lot of parameters are involved for both algorithms, in particular the target precision may play an important role, so that more extensive experiments may be needed to refine these statements.

Our goal here is to provide a simple but strong baseline with theoretical guarantees for recovery. Improving on it, as Fercoq and Qu [2016] did for LASSO, is an appealing research direction. Sharpness may be used for example to refine the heuristic strategy of the continuations steps.

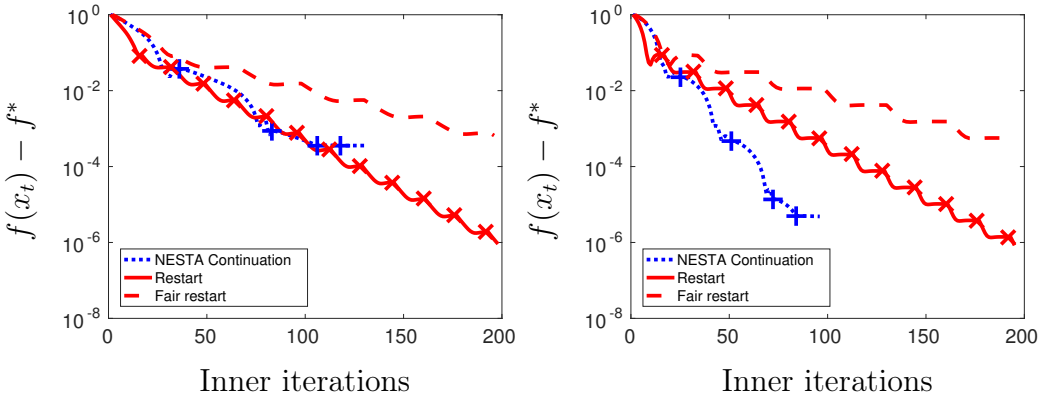

Figure 4-2 – Best restarted NESTA (solid red line) and overall cost of the practical restart schemes (dashed red line) versus NESTA with 5 continuation steps (dotted blue line) for a budget of 500 iterations. Crosses represent the restart occurrences. Left: $n = 200$. Right : $n = 300$.

Number of samples n	100	200	400
Time in seconds for $f(x_t) - f^* < 10^{-2}$	$5.07 \cdot 10^{-2}$	$3.07 \cdot 10^{-2}$	$1.66 \cdot 10^{-2}$

Table 4.1 – Time to achieve $\varepsilon = 10^{-2}$ by the best restart scheme for increasing number of samples n

Convergence rate and oversampling ratio

We now illustrate the theoretical results of Section 4.1.3 by running the practical scheme presented in Section 4.1.2 for increasing values of the oversampling ratio $\tau = n/s$. In Figure 4-3, we plot the best scheme found by the grid search, that approximates the optimal scheme, for a budget of $N = 500$ iterations. We use a fine grid of step size $h = 2$. Other algorithmic parameters remain unchanged: $x_0 = A^T b$ and $\varepsilon_0 = \|x_0\|_1$. We fix the dimension $p = 1000$ and either make n vary for a fixed sparsity $s = 20$ or make s vary for a fixed number of samples $n = 200$. In both cases we do observe an improved convergence for increasing oversampling ratio τ. Notice that linear convergence is observed even for $\tau < \log p$, that is, outside the recovery region. This suggests that sharpness may also hold in this case in the form of a Łojasiewicz inequality as mentioned in Sections 4.1.2 and 4.2.3.

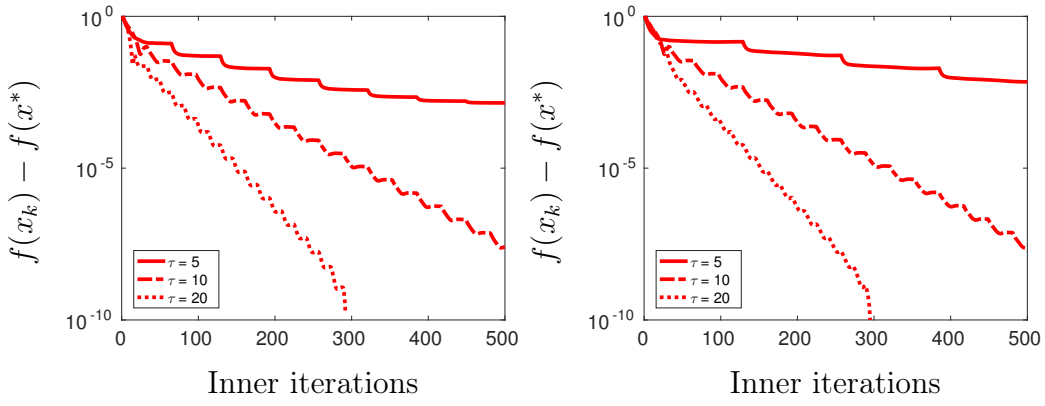

Figure 4-3 – Best restart scheme found by grid search for increasing values of the oversampling ratio $\tau = n/s$. Left : sparsity $s = 20$ fixed. Right : number of samples $n = 200$ fixed.

In table 4.1, we report the time needed to achieve an $\varepsilon = 10^{-2}$ precision for a fixed sparsity $s = 20$ and increasing number of samples $n = \{100, 200, 400\}$. While the cost of core operations (namely, matrix-vector multiplications) increases with n, the overall time required to reach convergence decreases. In other words, getting more samples increases the recovery performance at little or no computational cost.

4.4.2 Renegar's condition number and compressed sensing performance

Our theoretical results showed that Renegar's condition number measures the complexity for the exact recovery problem (4.1). However it does not a priori control convergence of the robust recovery problems defined in the introduction. This numerical section aims therefore at analyzing the relevance of this condition number for general recovery problems in the ℓ_1 case, assuming that their complexity corresponds roughly to that of checking optimality of a given point at each iteration, as mentioned in Section 4.2.3. We first describe how we approximate the value of $\mathcal{R}_{\mathcal{T}(x^*)}(A)$ as defined in (4.18) for a given original signal x^* and matrix $A \in \mathbb{R}^{n\times p}$. We then detail numerical experiments on synthetic data sets.

Computing $\mathcal{R}_{\mathcal{T}(x^*)}(A)$

The condition number $\mathcal{R}_{\mathcal{T}(x^*)}(A)$ appears here in upper bounds on computational complexities and statistical performances. In order to test numerically whether this quantity truly explains those features (as opposed to merely appearing in a wildly pessimistic bound), we explicitly compute it in numerical experiments.

To compute $\mathcal{R}_{\mathcal{T}(x^*)}(A)$, we propose a heuristic which computes $\sigma_{\mathcal{T}(x^*)}(A)$ in (4.11) and (4.12), the value of a non convex minimization problem over the cone of descent directions $\mathcal{T}(x^*)$. The closure of the latter is the polar of the cone generated by the subdifferential to the ℓ_1-norm ball at x^* [Chandrasekaran et al., 2012, §2.3]. Let $S \subset [\![1, p]\!]$ denote the support of x^* and $s = \mathbf{Card}(S)$. Then, with $u = \mathrm{sign}(x^*)$,

$$\mathcal{T}(x^*) = \mathrm{cone}\left\{z \in \mathbb{R}^p : z_S = u_S, z_{S^c} \in [-1,1]^{p-s})\right\}^\circ = \left\{z \in \mathbb{R}^p : \|z_{S^c}\|_1 \leq -u_S^T z_S = -u^T z\right\}.$$

Thus, $\sigma_{\mathcal{T}(x^*)}(A)$ is the square root of

$$\min_{z\in\mathbb{R}^p} z^T A^T A z \quad \text{s.t.} \quad \|z\|_2 = 1 \quad \text{and} \quad \|z_{S^c}\|_1 \leq -u^T z. \tag{4.24}$$

Let λ denote the largest eigenvalue of $A^T A$. If it were not for the cone constraint, solutions of this problem would be the dominant eigenvectors of $\lambda \mathbf{I} - A^T A$, which suggests a *projected power method* [Deshpande et al., 2014] as follows. Given an initial guess $z_0 \in \mathbb{R}^p$, $\|z_0\|_2 = 1$, iterate

$$\hat{z}_{k+1} = \mathrm{Proj}_{\mathcal{T}(x^*)}\left((\lambda \mathbf{I} - A^T A) z_k\right), \qquad z_{k+1} = \hat{z}_{k+1}/\|\hat{z}_{k+1}\|_2, \tag{4.25}$$

where we used the orthogonal projector to $\mathcal{T}(x^*)$,

$$\mathrm{Proj}_{\mathcal{T}(x^*)}(\tilde{z}) = \arg\min_{z\in\mathbb{R}^p} \|z - \tilde{z}\|_2^2 \quad \text{s.t.} \quad \|z_{S^c}\|_1 \leq -u^T z. \tag{4.26}$$

This convex, linearly constrained quadratic program is easily solved with CVX [Grant et al., 2001]. As can be seen from KKT conditions, this iteration is a generalized power

iteration [Luss and Teboulle, 2013; Journée et al., 2010]

$$z_{k+1} \in \arg\max_{z\in\mathbb{R}^p} z^T(\lambda \mathbf{I} - A^TA)z_k \quad \text{s.t.} \quad \|z\|_2 \leq 1 \quad \text{and} \quad \|z_{S^c}\|_1 \leq -u^Tz.$$

From the latter, it follows that $\|Az_k\|_2$ decreases monotonically with k. Indeed, owing to convexity of $f(z) = \frac{1}{2}z^T(\lambda\mathbf{I}-A^TA)z$, we have $f(z)-f(z_k) \geq (z-z_k)^T(\lambda\mathbf{I}-A^TA)z_k$. The next iterate $z = z_{k+1}$ maximizes this lower bound on the improvement. Since $z = z_k$ is admissible, the improvement is nonnegative and $f(z_k)$ increases monotonically.

Thus, the sequence $\|Az_k\|_2$ converges, but it may do so slowly, and the value it converges to may depend on the initial iterate z_0. On both accounts, it helps greatly to choose z_0 well. To obtain one, we modify (4.24) by smoothly penalizing the inequality constraint in the cost function, which results in a smooth optimization problem on the ℓ_2 sphere. Specifically, for small $\varepsilon_1, \varepsilon_2 > 0$, we use smooth proxies $h(x) = \sqrt{x^2+\varepsilon_1^2} - \varepsilon_1 \approx |x|$ and $q(x) = \varepsilon_2 \log(1+\exp(x/\varepsilon_2)) \approx \max(0, x)$. Then, with $\gamma > 0$ as Lagrange multiplier, we consider

$$\min_{\|z\|_2=1} \|Az\|_2^2 + \gamma \cdot q\left(u^Tz + \sum\nolimits_{i\in S^c} h(z_i)\right).$$

We solve the latter locally with Manopt [Boumal et al., 2014], itself with a uniformly random initial guess on the sphere, to obtain z_0. Then, we iterate the projected power method. The value $\|Az\|_2$ is an upper bound on $\sigma_{\mathcal{T}(x^*)}(A)$, so that we obtain a lower bound on $\mathcal{R}_{\mathcal{T}(x^*)}(A)$. Empirically, this procedure, which is random only through the initial guess on the sphere, consistently returns the same value, up to five digits of accuracy, which suggests the proposed heuristic computes a good approximation of the condition number. Similarly positive results have been reported on other cones by Deshpande et al. [2014], where the special structure of the cone even made it possible to certify that this procedure indeed attains a global optimum in proposed experiments. Similarly, a generalized power method was recently shown to converge to global optimizers for the phase synchronization problem (in a certain noise regime) [Boumal, 2016; Zhong and Boumal, 2017]. This gives us confidence in the estimates produced here.

Sparse recovery performance

We conduct numerical experiments in the ℓ_1 case to illustrate the connection between the condition number $\mathcal{R}_{\mathcal{T}(x^*)}(A)$, the computational complexity of solving (ℓ_1 recovery), and the statistical efficiency of the estimator (Robust ℓ_1 recovery). Importantly, throughout the experiments, the classical condition number of A will remain essentially constant, so that the main variations cannot be attributed to the latter.

We follow a standard setup, similar to some of the experiments by Donoho and Tsaig [2008]. Fixing the ambient dimension $p = 300$ and sparsity $s = \|x^*\|_0 = 15$, we let the number of linear measurements n vary from 1 to 150. For each value of n, we generate a random signal $x^* \in \mathbb{R}^p$ (uniformly random support, i.i.d. Gaussian entries, unit ℓ_2-norm) and a random sensing matrix $A \in \mathbb{R}^{n\times p}$ with i.i.d. standard Gaussian

entries. Furthermore, for a fixed value $\delta = 10^{-2}$, we generate a random noise vector $w \in \mathbb{R}^n$ with i.i.d. standard Gaussian entries, normalized such that $\|w\|_2 = \delta \|A\|_2$, and we let $b = Ax^* + w$. This is repeated 100 times for each value of n.

For each triplet (A, x^*, b), we first solve the noisy problem (Robust ℓ_1 recovery) with the L1-Homotopy algorithm ($\tau = 10^{-7}$) [Asif and Romberg, 2014], and report the estimation error $\|\hat{x} - x^*\|_2$. Then, we solve the noiseless problem (4.1) with L1-Homotopy and the TFOCS routine for basis pursuit ($\mu = 1$) [Becker, Candès and Grant, 2011]. Exact recovery is declared when the error is less than 10^{-5}, and we report the empirical probability of exact recovery, together with the number of iterations required by each of the solvers. The number of iterations of LARS [Efron et al., 2004] is also reported, for comparison. For L1-Homotopy, we report the computation time, normalized by the computation time required for one least-squares solve in A, as in [Donoho and Tsaig, 2008, Fig. 3], which accounts for the growth in n. Finally, we compute the classical condition number of A, $\kappa(A)$, as well as (a lower bound on) the cone-restricted condition number $\mathcal{R}_{\mathcal{T}(x^*)}(A)$, as per the previous section. As it is the computational bottleneck of the experiment, it is only computed for 20 of the 100 repetitions.

The results of Figure 4-4 show that the cone-restricted condition number explains both the computational complexity of (ℓ_1 recovery) and the statistical complexity of (Robust ℓ_1 recovery): fewer samples mean bad conditioning which in turn implies high computational complexity. We caution that our estimate of $\mathcal{R}_{\mathcal{T}(x^*)}(A)$ is only a lower bound. Indeed, for small n, the third plot on the left shows that, even in the absence of noise, recovery of x^* is not achieved by (Robust ℓ_1 recovery). Lemma 4.2.3 then requires $\mathcal{R}_{\mathcal{T}(x^*)}(A)$ to be infinite. But the computational complexity of solving (ℓ_1 recovery) is visibly favorable for small n, where far from the phase transition, problem ($\mathrm{P}_{A,\mathcal{T}(x)}$) is far from infeasibility, which is just as easy to verify as it is to certify that ($\mathrm{P}_{A,\mathcal{T}(x)}$) is infeasible when n is comfortably larger than needed. This phenomenon is best explained using a symmetric version of the condition number [Amelunxen and Lotz, 2014] (omitted here to simplify computations).

We also solved problem (ℓ_1 recovery) with interior point methods (IPM) via CVX. The number of iterations appeared mostly constant throughout the experiments, suggesting that the practical implementation of such solvers renders their complexity mostly data agnostic in the present setting. Likewise, the computation time required by L1-Homotopy on the noisy problem (Robust ℓ_1 recovery), normalized by the time of a least-squares solve, is mostly constant (at about 150). This hints that the link between computational complexity of (ℓ_1 recovery) and (Robust ℓ_1 recovery) remains to be fully explained.

4.5 Conclusion

We studied the geometry of sparse recovery problems around their solutions that control both computational efficiency of restart schemes or oracle-based algorithms for exact recovery, and statistical performance of the decoding procedures, either in the recovery threshold or in the sensitivity to noise. We generally show that

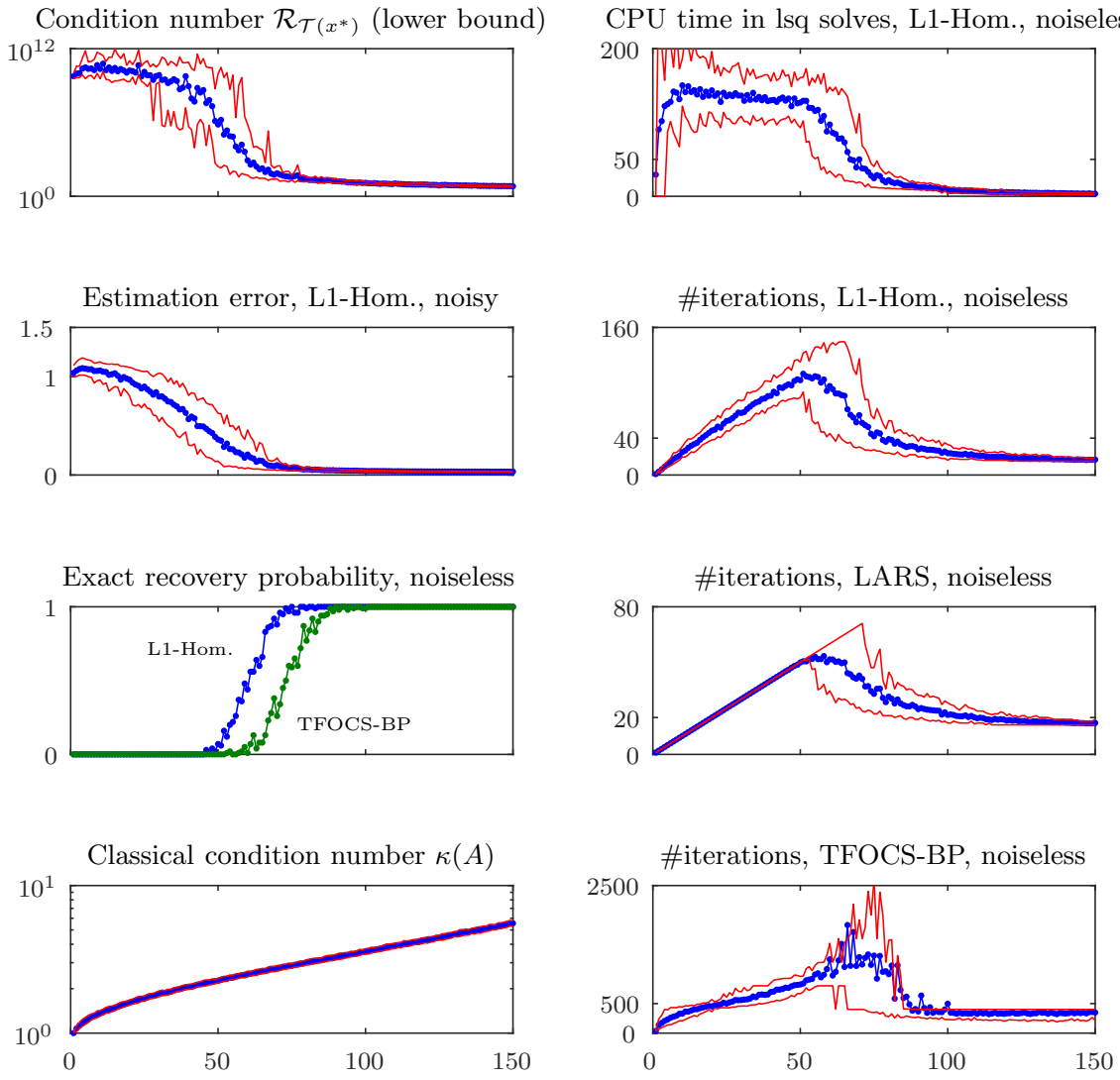

Figure 4-4 – We plot the cone-restricted condition number of A (upper left), explaining both the computational complexity of problem (ℓ_1 recovery) (right column) and the statistical complexity of problem (Robust ℓ_1 recovery) (second on the left). Central curves represent the mean (geometric mean in log-scale plots), red curves correspond to 10th and 90th percentile. We observe that high computing times (peaks in the right column) are directly aligned with instances where sparse recovery barely holds/fails (left), i.e. near the phase transition around $n = 70$, where the distance to feasibility for problem ($\mathrm{P}_{\mathrm{A},\mathcal{T}(x)}$) also follows a phase transition.

minimal conically restricted singular value of the coding matrix is the key quantity that describes the problem and can be seen as the worst case of Renegar's condition number for conic feasibility problems that certificate recovery. This analysis extends then to other sparse structures such as group sparsity or low rank matrices.

Several questions remain. First, sharpness constant is only given when exact recovery is ensured. As the ℓ_1 norm is always sharp, our analysis may extend to problems where exact recovery is not possible. In other words, if the decoding procedure is far from being efficient statistically, its complexity may also be small. Furthermore, for robust recovery, sharpness may also be stated. A finer analysis may help to link it with minimal conically restricted singular value on the tangent cone of the original vector which would give instance dependent complexity bounds. Finally a proper definition of sparse structures would ease the development of new compressed structures such as grouped vectors that we present in Chapter 6.

Appendix

4.A Practical optimal restart scheme

In Section 4.1.2 we quickly give optimal restart schemes in terms of a potentially non-integer clock. Following corollary details optimal scheme for an integer optimal clock.

Corollary 4.A.1. *Given a coding matrix $A \in \mathbb{R}^{n\times p}$ and an original signal $x^* \in \mathbb{R}^p$ such that sharpness bound (Sharp) is satisfied with $\gamma > 0$, running Algorithm Restart with ρ^* and $t = \lceil t^* \rceil$ where ρ^* and t^* are defined in (4.4) ensures that after $K \geq 1$ restarts, i.e. $N = K\lceil t^* \rceil$ total number of iterations,*

$$\|\hat{y}\|_1 - \|x^*\|_1 \leq \exp\left(-\frac{N\gamma}{2e\sqrt{p}+\gamma}\right)\varepsilon_0. \tag{4.27}$$

Proof. Denote $\delta = \lceil t^* \rceil - t^* \in [0, 1[$. As $\lceil t^* \rceil \geq t^*$ (4.3) is ensured for ρ^*. At the K^{th} restart, $N = K(t^* + \delta)$, and

$$\|\hat{y}\|_1 - \|x^*\|_1 \leq e^{-K}\varepsilon_0 = \exp(-N/(t^* + \delta)) \leq \exp(-N/(t^* + 1)).$$

Replacing t^* by its value gives the result. ∎

4.B Remark on sparsity inducing norms

We quickly discuss the framework of Juditsky et al. [2014] for sparsity inducing norms and show that it can be simplified. We first recall the definition.

Definition 4.B.1. (Sparsity structure [Juditsky et al., 2014]) *A sparsity structure on a Euclidean space E is defined as a norm $\|\cdot\|$ on E, together with a family $\mathcal{P}$ of linear maps of E into itself, satisfying three assumptions:*

1. *Every $P \in \mathcal{P}$ is a projector, $P^2 = P$,*
2. *Every $P \in \mathcal{P}$ is assigned a weight $\nu(P) \geq 0$ and a linear map $\bar{P}$ on E such that $P\bar{P} = 0$,*
3. *For any $P \in \mathcal{P}$ and $u, v \in E$, one has*

$$\|P^*u + \bar{P}^*v\|_* \leq \max(\|u\|_*, \|v\|_*),$$

where $\|\cdot\|_*$ *is the dual norm of* $\|\cdot\|$ *and* P^* *is the conjugate mapping of the linear map* P*.*

The last condition in Definition 4.B.1 is arguably the least intuitive and following Lemma connects it with the more intuitive notion of decomposable norm.

Lemma 4.B.2. *Condition* (3) *above, which reads*

$$\|P^*u + \bar{P}^*v\|_* \leq \max(\|u\|_*, \|v\|_*),$$

for any $u, v \in E$*, is equivalent to*

$$\|w\| \geq \|Pw\| + \|\bar{P}w\|,$$

for any $w \in E$*.*

Proof. Denote $f : (u, v) \to \|P^*u + \bar{P}^*v\|_*$ and $g : (u, v) \to \max(\|u\|_*, \|v\|_*)$. Since f and g are non-negative, continuous convex functions, $f^2/2$ and $g^2/2$ are also convex continuous and following equivalences hold

$$f \leq g \quad \Leftrightarrow \quad \frac{f^2}{2} \leq \frac{g^2}{2} \quad \Leftrightarrow \quad \left(\frac{f^2}{2}\right)^* \geq \left(\frac{g^2}{2}\right)^*,$$

using that for a convex continuous function h, $h^{**} = h$. Now combining the conjugacy result for squared norm [Boyd and Vandenberghe, 2004, Example 3.27] showing that the conjugate of a squared norm $\|x\|^2/2$ is the squared conjugate norm $\|x\|_*^2/2$, with the result in [Rockafellar, 2015, Th. 16.3], we get

$$\left(\frac{f^2}{2}\right)^* (s, t) = \inf_w \{\|w\|^2/2 : Pw = s, \bar{P}w = t\},$$

where the infimum is $+\infty$ if the constraints are infeasible. Then the dual of the norm g is $(s, t) \to \|s\| + \|t\|$ therefore condition (3) is equivalent to

$$\inf_w \{\|w\| : Pw = s, \bar{P}w = t\} \geq \|s\| + \|t\|,$$

for any $s, t \in E$, which reads

$$\|w\| \geq \|Pw\| + \|\bar{P}w\|,$$

for any $w \in E$. ∎

Part II

Machine learning problems with partitioning structure

Chapter 5

Introduction

Machine learning

Machine learning is a recent field of research that attempts to give to a computer the ability to perform predictive analysis of data. It stems from statistics which model the data and optimization which makes the computer find the best parameters of the model. It has been applied to numerous fields such as computer vision, natural language processing, bio-informatics, robotics, speech processing and economics. Machine learning can be applied for example to predict antigens concentration in blood after a medical treatment. Another classical application is image classification, where the computer must distinguish pictures of humans, animals or objects.

Formally, data consist of objects $x \in \mathcal{X}$ and their attributes $y \in \mathcal{Y}$ that the computer must predict. To perform predictive analysis, the computer has generally access to the representation of the object by d features, such that throughout this part $\mathcal{X} = \mathbb{R}^d$. Attributes can be quantitative values, i.e. $\mathcal{Y} = \mathbb{R}$, as in prediction of antigens concentration, or qualitative values, $\mathcal{Y} = \{1, \dots K\}$ that encode K classes of objects, as in image classification.

Several settings exist to analyze data depending on the information that a computer can have access. In this thesis, we study supervised learning problems where a batch of training samples is available. It consists in n pairs $(x_1, , y_1), \dots, (x_n, y_n)$ of objects and their attributes that the computer uses to fit a model on the data.

Big data challenges

Machine learning appeals a growing interest of researchers because of the large amount of data available in applications that the prediction task can benefit from. By large, one refers first to the number n of training samples like the number of documents in a topic classification task. The larger is n, the more performing is naturally the computed model.

Big data can also concern the number d of features that describe the data. For example in genomics, features are frequency of mutations of genes and the number of discriminative genes in ADN is around 3 millions. Similarly, the more features one can have access to describe the data, the more information the computer can use.

Finally, machine learning problems may treat large number K of tasks simultaneously. For example, in computer vision, a common task is to identify elements of a picture such as a cat or a car. Each element requires a specific classification task and data sets like ImageNet [Deng et al., 2009] of 1000 classes exist to describe the variety of possible elements in a picture. Here again a large number K of tasks may help because similar tasks can share information to improve overall performance.

While this big data setting helps prediction performance, it also increases complexity of the task. First, it rises new computational challenges for the optimization procedure but the resulting model can also be more difficult to interpret. Learning procedures able to perform prediction and simplify the problem are therefore of interest for the user. They not only help interpretation but also increase generalization performance.

Outline of this part

In this part, we study therefore learning procedures that simultaneously solve a prediction problem and reduce its complexity by grouping either features in Chapter 6, samples in Chapter 7 or tasks in Chapter 8. They mix classical supervised learning procedures and partitioning problems. The algebraic tools and optimization procedures are essentially the same for all three settings. We present them in Sections 5.2 and 5.3. A general perspective of our approach is given in Section 5.1. Following chapters detail then each application and specific computations.

5.1 Learning in the data cube

For a general supervised machine learning problem involving n samples with d features to solve K tasks, supervised learning methods have at their disposition n input data points $x_1, \dots, x_n \in \mathbb{R}^d$ with their corresponding outputs for K tasks $y_1, \dots, y_n \in \mathbb{R}^K$. Linear methods aim then at finding K vectors of prediction $w_1, \dots w_K \in \mathbb{R}^d$ that maps inputs to outputs as

$$y_{ik} \approx w_k^T x_i, \quad \text{for } k \in \{1, \dots, K\} \text{ and } i \in \{1, \dots, n\}$$

which reads compactly

$$Y \approx XW,$$

where $Y = (y_1, \dots, y_n)^T \in \mathbb{R}^{n \times K}$, $X = (x_1, \dots, x_n)^T \in \mathbb{R}^{n \times d}$ and $W = (w_1, \dots, w_k) \in \mathbb{R}^{d \times K}$. A loss function is then used to measure the accuracy error of a given set of predictors W, as detailed for regression and classification in the following chapters. Its minimization on the training data gives a candidate predictor for future samples.

Notice that to perform prediction we assume that all samples share the same predictor. However, to account for diversity in the samples, this assumption can be relaxed by searching for more than one predictor as detailed in Chapter 7. Diverse outputs are then delivered by the machine depending on the prediction variable

chosen. This means that our prediction variable can generally be seen as a tensor

$$\mathcal{W} = (w_{jki})_{j\in\{1,\dots d\},\, k\in\{1,\dots,K\},\, i\in\{1,\dots,n\}} \in \mathbb{R}^{d\times K\times n}$$

whose coefficients along the third direction are constrained to be equal for classical prediction which leads to the matrix of prediction W presented before. The three dimensions of this tensor dress the data cube as introduced by Harchaoui [2013]. In this part we study grouping of either features, samples or tasks by constraining coordinates of the tensor $\mathcal{W}$ along the different directions it defines in the data cube. Precisely, we constraint the coefficients to be equal or closed to each other in some groups defined by partitions.

5.2 Partitioning problems

5.2.1 Representation of partitions

We present classical algebraic tools to partition p items, where, in our following settings, $p = d$ for features, $p = n$ for samples or $p = K$ for tasks. First, recall the definition of partitions.

Definition 5.2.1. ***Partitions*** *A collection G of subsets of $\{1,\dots,p\}$ is a partition of $\{1,\dots,p\}$ if for any $g, g' \in G\times G$, $g \neq g'$ implies $g\cap g' = \emptyset$ and if $\bigcup_{g\in G} g = \{1,\dots,p\}$.*

Partitions can be partially ordered as detailed in Section 6.4.2. They possess a lattice structure [Fujishige, 2005] that can be used for combinatorial optimization [Topkis, 1978; Amini et al., 2009].

Here we use simple algebraic tools to represent them. First, for a partition $G = (g_1,\dots,g_Q)$ of a set $\{1,\dots,p\}$ of p elements in Q groups, one defines assignment matrices $Z \in \{0,1\}^{p\times Q}$ that assigns items into the groups $g_1,\dots,g_Q$ by

$$Z_{iq} = \begin{cases} 1 & \text{if} \quad i \in g_q, \\ 0 & \text{otherwise.} \end{cases}$$

Observe that a partition G into Q groups $g_1,\dots g_Q$ is independent of the ordering of the groups, namely, $g_{\pi(1)},\dots g_{\pi(Q)}$, where π is a permutation of $\{1,\dots,Q\}$, describes as well G. Consequently a partition can be encoded by several assignment matrices, these are identical up to a permutation of their columns defining the groups.

A binary matrix $Z \in \{0,1\}^{p\times Q}$ describes a partition G of $\{1,\dots,p\}$ into Q groups, if and only if it satisfies $Z\mathbf{1} = \mathbf{1}$ as it encodes the fact that each element belongs to exactly one group. Since groups of a partition are disjoints, columns of assignment matrices are orthogonal. Their squared Euclidean norm and ℓ_1 norm are equal to the size of the groups they represent, i.e. $\|Z_q\|_2^2 = \|Z_q\|_1 = Z^T\mathbf{1} = \mathbf{Card}(g_q)$, where Z_q is the q^{th} column of an assignment matrix Z of a partition $G = (g_1,\dots,g_Q)$. Combining two previous comments, we conclude that size of the groups are the squared singular values of the assignment matrix., i.e. $Z^TZ = \mathbf{diag}(s)$ where $s = (\mathbf{Card}(g_1,\dots,\mathbf{Card}(g_Q))$ encodes the size of the groups $g_1,\dots,g_Q$ that Z represents.

Another way to represent a partition $G = (g_1, \ldots, g_Q)$ of $\{1, \ldots, p\}$ into Q groups is to use its equivalence matrix $N \in \{0,1\}^{p \times p}$ that encodes if pair of items belong to the same group as

$$N_{ij} = \begin{cases} 1 & \text{if} \quad (i,j) \in (g_q \times g_q) \\ 0 & \text{otherwise.} \end{cases}$$

They can be derived from assignment matrices Z of the partition G as $N = ZZ^T$. Contrary to assignment matrices, there exists only one equivalence matrix per partition. In other words, partitions are uniquely defined by pairwise relationships of the items. Equivalence matrices are notably used for finding minimal cuts of graph, which partitions its vertices. Resulting problem is then convex but can lead to unsatisfactory solutions as it may only separate one edge to the others [Von Luxburg, 2007].

To circumvent this problem, balanced cuts penalize the resulting partitions by the size of the groups. This leads to the definition of normalized equivalence matrices $M \in \mathbb{R}^{p \times p}$, that encode a partition $G = \{g_1, \ldots, g_Q\}$ of $\{1, \ldots, p\}$ as

$$M_{ij} = \begin{cases} 1/\operatorname{\mathbf{Card}}(g_q) & \text{if} \quad (i,j) \in (g_q \times g_q) \\ 0 & \text{otherwise.} \end{cases}$$

In terms of assignment matrices, these read $M = Z(Z^TZ)^\dagger Z^T$, where $A^\dagger$ denotes the pseudo-inverse of a matrix A, here $(Z^TZ)^\dagger = \operatorname{\mathbf{diag}}(s^\dagger)$, where $s_q^\dagger = 1/\operatorname{\mathbf{Card}}(g_q)$ if g_q is non-empty and $s_q^\dagger = 0$ otherwise. Normalized equivalence matrices are orthonormal projectors, i.e. $M^2 = M$ and $M^T = M$. While normalized equivalence matrices may better formulate a task, they lead to non-convex problems such as k-means.

5.2.2 K-means problem

A well-known and appealing partitioning problem is the clustering of points into groups of points closed to each other. Its common example is the k-means problem [Steinhaus, 1956; MacQueen et al., 1967] that seeks to partition p points $x_1, \ldots, x_p \in \mathbb{R}^d$ into Q clusters that minimize distance between points and centers $c_1, \ldots, c_Q$ of the clusters. Several distances can be used [Dhillon et al., 2004], the original one is the Euclidean distance which leads to following problem

$$\text{minimize} \sum_{q=1}^{Q} \sum_{i \in g_q} \|x_i - c_q\|_2^2,$$

in the partition $G = (g_1, \ldots, g_Q)$ of $\{1, \ldots, p\}$ and the centroids $c_1, \ldots, c_Q$. By using assignment matrices $Z \in \{0,1\}^{p \times Q}$ to represent partitions, the k-means problem reads

$$\text{minimize} \|X - ZC\|_F^2 \tag{5.1}$$

in variables $C = (c_1, \ldots, c_Q)^T \in \mathbb{R}^{Q \times d}$ and Z, where $X = (x_1, \ldots, x_p)^T \in \mathbb{R}^{p \times d}$ is the matrix of data points. In following chapters, we develop formulations of our predic-

tion problems that group features, samples or tasks into Q groups using assignment matrices Z such that constraint set takes the form

$$\{W = VZ, Z \in \{0,1\}^{p\times Q}\}$$

where p is the number of items to be grouped, W represents predictors with potentially redundant coefficients and V represents the predictors without redundancy. Dimensions of W and V depend on the setting. Assignment matrices make the constraint set non-convex but projection on it reads (5.1). Therefore projected gradient presented in next section can be approximately performed.

K-means problem (5.1) can be simplified by minimizing in the matrix of centroids which leads to

$$\begin{array}{ll} \text{minimize} & \|U - Z(Z^TZ)^{-1}Z^TU\|_F^2 \\ \text{subject to} & Z \in \{0,1\}^{d\times Q}, \quad Z\mathbf{1} = \mathbf{1}, \end{array} \tag{5.2}$$

in variable Z, where we recognize normalized equivalence matrices of partitions $M = Z(Z^TZ)^{-1}Z^T$. In our prediction problems, when using a squared loss, empirical loss minimization can be performed analytically. It remains a partitioning problem in terms of normalized equivalence matrices that can be solved using conditional gradient presented in next section. Core inner step of this algorithm, namely linear minimization oracle, amounts indeed to the k-means problem (5.2), as detailed in following chapters.

5.3 Optimization on non-convex sets

By imposing partitioning structure on a classical learning task we introduce non-convex constraints in the optimization procedure. Formally, we face a problem of the form

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in C, \end{array} \tag{5.3}$$

in $x \in \mathbb{R}^d$, where f is a L-smooth convex function and C is a closed set. In this section we review optimization strategies for such problems.

5.3.1 Projected gradient descent

Projected gradient scheme initialized on $x_0 \in C$ for (5.3) reads

$$x_{t+1} = P_C(x_t - \gamma_t \nabla f(x_t)),$$

where γ_t is the step-size and $P_C(x) \in \operatorname{argmin}_{y\in C} \|x - y\|_2$ is the projection on set C.

If C is convex, projection is uniquely defined and a constant step-size $\gamma_t = \frac{1}{L}$ ensures convergence to a solution of (5.3) at rate $O(1/t)$ [Nesterov, 2013*b*]. When C is non-convex convergence is no more guaranteed without further assumptions on the problem.

Yet, projected gradient schemes offer fast and good algorithmic solutions for nu-

merous models. In some cases their convergence can even be stated. A well-known example is the Iterative Hard Thresholding algorithm [Blumensath and Davies, 2009] used in compressed sensing to decode a given sparse signal from linear observations of it. Provided that linear observations satisfy the restricted isometric property [Candès and Tao, 2005], Iterative Hard Thresholding is shown to approximately recover the sparse signal. Such analysis was extended to low rank estimation of matrices [Tanner and Wei, 2013] or tensors [Rauhut et al., 2017]. Key ingredients are the structure of the constraint set, namely a union of subspaces and the restricted strong convexity property that the function f satisfies [Jain et al., 2014].

In our settings, projected gradient descent offer fast and scalable resolutions of our problems. However their analysis is delicate, since, even if the feasible set is a union of subspaces, the projection step is generally only approximated, as k-means is a NP-hard problem [Mahajan et al., 2012]. Yet, in the case of regression grouping features, the k-means steps are performed on scalar values and can therefore be solved exactly by dynamic program. This allows us in Chapter 6 to adapt the analysis of classical Iterative Hard Thresholding algorithm in our setting

5.3.2 Convex relaxation

A more classical approach to solve non-convex problems of the form (5.3) is to approximate the non-convex set by a convex one. This ensures then efficient optimization of the resulting problem. The tightest relaxation consists in considering the convex hull of the original set which leads to

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \text{hull}(C), \end{array}$$

in $x \in \mathbb{R}^d$. Projected gradient descent may then be delicate as it would require to build the convex hull and the projection itself may be costly. In such cases, the conditional gradient method, a.k.a. Frank-Wolfe [Frank and Wolfe, 1956; Jaggi, 2013], presented in Algorithm 5, can circumvent the problem. It uses a linear minimization oracle (5.4) to produce a sequence of feasible iterates that converge to a solution x^* of the problem at a rate $1/t$, where t is the number of iterations. The linear minimization oracle also provides an estimated gap (5.5) for free, since, by definition of the oracle and convexity of f,

$$\langle x_t - s_t, \nabla f(x_t) \rangle \geq \langle x_t - x^*, \nabla f(x_t) \rangle \geq f(x_t) - f(x^*).$$

Algorithm 5 Conditional gradient algorithm for constrained problem (5.3)

Inputs: Initial point $x_0 \in \text{hull}(C)$, target precision ε
for $t = 0, \ldots$ **do**
Solve linear minimization oracle

$$s_t = \underset{s \in \text{hull}(C)}{\text{argmin}} \langle \nabla f(x_t), s \rangle \tag{5.4}$$

Get estimated gap

$$\Delta_t = \langle x_t - s_t, \nabla f(x_t) \rangle \tag{5.5}$$

if $\Delta_t \leq \varepsilon$ **then** Stop **end if**
Set

$$x_{t+1} = x_t + \frac{2}{t+2}(s_t - x_t)$$

end for
Output: $\hat{x} = x_t$

The key observation is that, in our problems, we do have access to a linear minimization oracle that amounts to a k-means problem.

Chapter 6

Grouping features for prediction with partitioning constraints

Chapter Abstract

In a prediction problem, grouping features can improve performance, robustness and interpretation of the results. Here we propose to find the best partition of the features for a task by constraining the prediction vector to have a small number of values. We formulate our model for classification and regression and present algorithmic schemes to tackle it. First, we develop a convex relaxation for squared losses using conditional gradient descent to handle the underlying combinatorial problem. Then, we propose a projected gradient scheme that amounts to iteratively cluster the features at each gradient step. We provide a theoretical analysis of this method to recover the information needed for prediction, based on a union of subspaces interpretation of the partitioning structure. We extend these results to combine sparsity and grouping constraints, and develop a new projection algorithm on the set of grouped sparse vectors. Numerical experiments illustrate the performance of our algorithms on synthetic and real data.

Introduction

In a prediction problem, getting a compressed representation of the information needed for the task has been extensively studied to improve prediction performance. Numerous models have been developed to select few features for the task (see e.g. [Tang et al., 2014]). In particular an extensive literature has been presented to tackle the problem by enforcing sparsity on the prediction vector (see [Bach et al., 2012]). Here we rather focus on the problem of grouping features, which has various applications. For example, in text classification this amounts to group words that have the same meaning for the task (see e.g. [Gupta et al., 2009] and references therein). In biology, this can be used to retrieve groups of genomes that have the same impact on a disease (see e.g. [Segal et al., 2003; Balding, 2006]). More generally this

approach can be seen as a supervised quantization of the feature space (see e.g. [Nova and Estévez, 2014] and references therein).

The idea of grouping features to reduce dimensionality of the problem is of course not new. Hastie et al. [2001] used for example supervised learning methods to select group of predictive variables formed by hierarchical clustering. Several models also developed mutual information-based algorithms to remove redundant features, e.g. [Yu and Liu, 2003; Song et al., 2013; Peng et al., 2005]. More recently, regularizers were developed to enforce grouped vectors [Bondell and Reich, 2008; She et al., 2010; Petry et al., 2011]. In particular, Bach et al. [2012] analyzed geometrical properties induced by convex relaxations of submodular functions that lead to group structures. This geometrical perspective was also investigated by Bühlmann et al. [2013], who showed recovery performance of group norms induced by hierarchical clustering methods based on canonical correlations. Finally Shen and Huang [2010] developed an homotopy method to extract homogeneous subgroups of predictors.

In this chapter, we study a simple approach to the problem : while sparsity enforces a small number of non-zero coefficient of the prediction vector, we enforce a small number of values. This naturally induces groups of features that share the same weight for the prediction. We formulate our approach for regression and classification tasks in Section 6.1, by using assignment matrices to link features to the representative coefficient of their group. This leads to a non-convex problem that we tackle with several algorithmic schemes.

First, in Section 6.2, we analyze a convex relaxation of our model when a squared loss is used to measure the prediction error. In that case, the underlying combinatorial problem can be isolated such that we can use a conditional gradient algorithm, a.k.a. Frank-Wolfe algorithm [Frank and Wolfe, 1956; Jaggi, 2013], whose core inner step amounts to solving a clustering problem. Then, in Section 6.3 we present a simple projected gradient scheme similar to the Iterative Hard Thresholding (IHT) [Blumensath and Davies, 2009] algorithm used in compressed sensing. While constraints are non-convex, projection on the feasible set also reduces to a clustering subproblem. For both schemes, the clustering problem can be solved exactly with dynamic programming [Bellman, 1973; Wang and Song, 2011] for regression where it amounts to a k-means problem in one dimension, or approximated efficiently with k-means++ [Arthur and Vassilvitskii, 2007] for classification.

We analyze the performance of the projected gradient scheme to recover a vector that generates the observations in Section 6.4. To this end, we detail the geometry induced by the partitioning constraints and demonstrate that, for regression with a least-square penalty and a sufficient number of observations, the projected gradient scheme can be seen as a fixed point algorithm. Although our structure is similar to sparsity, we show that imposing a grouped structure, while helping interpretability, does not allow us to significantly reduce the number of observations to retrieve the original vector, as in the sparse case for example.

We also extend the application of the projected gradient scheme to both select and group features in Section 6.5 by developing a new dynamic program that gives the exact projection on the set of sparse and clustered vectors.

Finally, numerical experiments illustrate the performance of the algorithmic schemes

on both synthetic and real datasets involving large corpora of text from movie reviews. The use of k-means steps makes our approach fast and scalable while comparing favorably with standard benchmarks and providing meaningful insights on the data structure.

6.1 Problem Formulation

We first present our framework for linear regression tasks and then extend its application to classification.

6.1.1 Regression with grouped features

Given n observations $y_1, \dots, y_n \in \mathbb{R}$ from data points $x_1, \dots, x_n \in \mathbb{R}^d$, linear regression aims at finding a regression vector $w \in \mathbb{R}^d$ that fits the data such that

$$y_i \approx w^T x_i, \quad \text{for all} \quad i = 1, \dots, n.$$

Parameter w then serves to predict future observations y of new data points x. To assess the quality of a prediction vector w, one defines a loss function ℓ that measures its accuracy error $\ell(w^T x, y)$ on a sample (x, y). A common choice of loss, that we investigate here, is the squared loss $\ell_{\text{square}}(w^T x, y) = \frac{1}{2}(w^T x - y)^2$, but several others exist in the literature, see Hastie et al. [2008]. A classical approach to compute a linear regression vector is then to minimize the empirical loss function

$$L(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(w^T x_i, y_i).$$

In order to prevent the computed prediction parameters from over-fitting the given set of samples, one often adds a regularizer $R(w)$ of the regression vector to the minimization problem. This notably reduces the effect of noise or outliers in the data. For example, one can use the squared Euclidean norm of the regression vector, i.e. $R_{\text{square}}(w) = \frac{1}{2}\|w\|^2$. Candidate regression parameters are then given by the minimization problem

$$\text{minimize } L(w) + \lambda R(w) \tag{6.1}$$

in variable $w \in \mathbb{R}^d$, where $\lambda \geq 0$ is a regularization parameter.

Structural information on the task can then be added. For example, one can enforce the regression vectors w to be sparse, i.e. to have few non-zeros coefficients. Here we rather enforce the regression vectors w to have at most Q values $v_1, \dots, v_Q$. Each coefficient v_q is assigned to a group of features g_q such that regression vectors w define partitions $G = \{g_1, \dots, g_Q\}$ of the features. We encode it by an assignment matrix $Z \in \{0, 1\}^{d \times Q}$, whose rows index the features and columns index the groups, such that

$$Z_{iq} = \begin{cases} 1 & \text{if} \quad i \in g_q \\ 0 & \text{otherwise.} \end{cases}$$

These are presented in Section 5.2. A regression vector w satisfying the constraints can then be described by an assignment matrix Z and the prediction weights $v_1, \dots, v_Q$ such that $w_i = \sum_{q=1}^{Q} Z_{iq} v_q = (Zv)_i$. Therefore linear regression enforcing Q groups of features reads

$$\begin{array}{ll} \text{minimize} & L(w) + \lambda R(w) \\ \text{subject to} & w = Zv, \quad Z \in \{0,1\}^{d \times Q}, \quad Z\mathbf{1} = \mathbf{1} \end{array} \tag{6.2}$$

in variables $w \in \mathbb{R}^d$, $v \in \mathbb{R}^Q$ and Z, where $\lambda \geq 0$ is a regularization parameter.

We detail the geometry of the non-convex set defined by the partitions of the features in Section 6.4.2 and extend now this formulation to classification tasks. Before notice that affine regression problems that seek for a regression vector $w \in \mathbb{R}^d$ and an intercept $b \in \mathbb{R}$ such that $y \approx w^T x + b$ can be treated similarly. It suffices to add a constant feature equals to one to data points x and to consider the resulting problem in dimension $d+1$. In this case regularization function R and partitioning constraints apply only on the first d dimensions of the resulting problem.

6.1.2 Classification with grouped features

Numerous models have been proposed for classification, we refer the interesting reader to Hastie et al. [2008] for a detailed presentation. Here we briefly present one of them, namely one-vs-all linear classification, in order to focus on the optimization problem that will be constrained to group features. In classification, data points $x_1, \dots, x_n \in \mathbb{R}^d$ belong to one of K classes, which can be encoded by binary vectors $y_i \in \{-1, 1\}^K$ such that $y_{ik} = 1$ if i^{th} point belongs to class k and -1 otherwise. One-vs-all linear classification aims then at computing hyperplanes defining regions of space where points are more likely to belong to a given class. Such hyperplanes are defined by their normals $w_1, \dots, w_K$, forming a matrix of linear classifiers $W \in \mathbb{R}^{d \times K}$ whose classification error on a sample (x, y) is measured by a loss $\ell(W^T x, y)$ such as the squared loss $\ell_{\text{square}}(W^T x, y) = \frac{1}{2}\|W^T x - y\|_2^2$. One searches then to minimize the empirical loss function

$$L(W) = \frac{1}{n} \sum_{i=1}^{n} \ell(W^T x_i, y_i).$$

As for regression, a regularizer $R(W)$ can be added on the linear classifiers such as their squared euclidean norm $R_{\text{square}}(W) = \frac{1}{2} \sum_{k=1}^{K} \|w_k\|_2^2 = \frac{1}{2}\|W\|_F^2$. Candidate classification parameters are then given by solving

$$\text{minimize } L(W) + \lambda R(W) \tag{6.3}$$

in variable $W \in \mathbb{R}^{d \times K}$, where $\lambda \geq 0$ is a regularization parameter.

To group features, we will enforce the classifiers to share the same partition of their coefficients. Namely, if this partition is encoded by an assignment matrix Z and $v_k = (v_{1k}, \dots, v_{Qk})$ represent the Q different coefficients of the k^{th} linear classifier w_k,

then $w_k = Zv_k$. Linear classification enforcing Q groups of constraints then reads

$$\begin{array}{ll} \text{minimize} & L(W) + \lambda R(W) \\ \text{subject to} & W = ZV, \quad Z \in \{0,1\}^{d \times Q}, \quad Z\mathbf{1} = \mathbf{1} \end{array} \tag{6.4}$$

in variables $W \in \mathbb{R}^{d \times K}$, $V \in \mathbb{R}^{Q \times K}$ and Z, where $\lambda \geq 0$ is a regularization parameter. Observe that constraints in (6.4) are essentially the same as the ones in (6.2), except that these are formulated on matrices. However this simple difference will have important algorithmic implications. Notice that for binary classification, a vector of labels of dimension one is sufficient to encode the class information, such that binary classification reduces to a problem of the form (6.2). As for regression, this setting can be applied to compute affine hyperplanes by extending the problem in $d+1$ dimension and by applying regularization and constraints only on the first d dimensions.

6.2 Convex relaxation

We now present a first optimization strategy for solving prediction problems (6.2) and (6.4) that group features, whose general formulation is

$$\begin{array}{ll} \text{minimize} & L(W) + \lambda R(W) \\ \text{subject to} & W = ZV, \quad Z \in \{0,1\}^{d \times Q}, \quad Z\mathbf{1} = \mathbf{1} \end{array} \tag{P}$$

in variables $W \in \mathbb{R}^{d \times K}$, $V \in \mathbb{R}^{Q \times K}$ and Z, where L and R are respectively the loss and the regularizer of the problem and $\lambda \geq 0$ is a regularization parameter. Regression and binary classification cases corresponds to $K = 1$ and multiclassification to $K > 1$. The difficulty of problem (P) lies in its underlying combinatorial nature that we isolate in the case of a squared loss and squared regularizer. Then, we propose a convex relaxation of the resulting problem which amounts to optimize on the convex hull of the set of constraints by using a conditional gradient algorithm.

6.2.1 Simplified formulation for squared loss

The squared loss has the advantage to provide analytic solutions for prediction problems in variables W, V. By replacing $W = ZV$, the objective of problem (P) in the remaining variables Z, V reads

$$\begin{aligned} L_{\text{square}}(ZV) + \lambda R_{\text{square}}(ZV) &= \frac{1}{2n} \sum_{i=1}^{n} \|y_i - (ZV)^T x_i\|_2^2 + \frac{\lambda}{2} \|ZV\|_F^2 \\ &= \frac{1}{2n} \mathbf{Tr}(V^T Z^T X^T X Z V) + \frac{\lambda}{2} \mathbf{Tr}(V^T Z^T Z V) \\ &\quad - \frac{1}{n} \mathbf{Tr}(Y^T X Z V) + \frac{1}{2n} \mathbf{Tr}(Y^T Y), \end{aligned}$$

where $\lambda \geq 0$ is a regularization parameter, $X = (x_1, \ldots, x_n)^T \in \mathbb{R}^{n \times d}$ is the matrix of data points and $Y = (y_1, \ldots, y_n)^T \in \mathbb{R}^{n \times K}$ is the matrix of labels or the vector of observations in the regression case. Assume first Z to be full rank, then minimization in V leads to

$$\begin{aligned}\min_V L_{\text{square}}(ZV) + \lambda R_{\text{square}}(ZV) &= \frac{1}{2n} \mathbf{Tr}\left(Y^T \left(\mathbf{I} - XZ(Z^T X^T XZ + \lambda n Z^T Z)^{-1} Z^T X^T\right) Y\right) \\ &= \frac{1}{2n} \mathbf{Tr}\left(Y^T \left(\mathbf{I} + \frac{1}{n\lambda} XZ(Z^T Z)^{-1} Z^T X^T\right)^{-1} Y\right),\end{aligned}$$

where we simplified first expression by using the Sherman-Woodbury-Morrison formula. If Z is not full rank, some of its columns are null (some of the groups it represents are empty). Previous computations can then be performed by replacing Z by $\tilde{Z}$ defined with the non-zero columns of Z, such that $\tilde{Z}$ is full rank and $\tilde{Z}(\tilde{Z}^T \tilde{Z})^{-1} \tilde{Z}^T = Z(Z^T Z)^\dagger Z^T$, where $A^\dagger$ is the pseudo-inverse of A. Overall, we therefore get

$$\min_V L_{\text{square}}(ZV) + \lambda R_{\text{square}}(ZV) = \frac{1}{2n} \mathbf{Tr}\left(Y^T \left(\mathbf{I} + \frac{1}{n\lambda} XZ(Z^T Z)^\dagger Z^T X^T\right)^{-1} Y\right)$$

Resulting objective is convex in $Z(Z^T Z)^\dagger Z^T$ where we recognize the normalized equivalence matrices of partitions as presented in Section 5.2. The optimization problem then reads

$$\begin{array}{ll} \text{minimize} & \mathbf{Tr}\left(Y^T \left(\mathbf{I} + \frac{1}{n\lambda} XMX^T\right)^{-1} Y\right) \\ \text{subject to} & M \in \mathcal{M}. \end{array} \tag{6.5}$$

in variable M where $\mathcal{M} = \{M = Z(Z^T Z)^\dagger Z^T,\ Z \in \{0,1\}^{d \times Q},\ Z\mathbf{1} = \mathbf{1}\}$ is the set of normalized equivalence matrices for partitions of $\{1, \ldots, d\}$ into Q groups. The resulting problem is still non-convex due to the combinatorial nature of the set of normalized equivalence matrices. However one can then relax the problem by optimizing on its convex hull as presented in Section 5.3.2.

6.2.2 Conditional gradient algorithm

We detail the linear minimization oracle used by Frank-Wolfe Algorithm 5 in this setting. Denote the objective function of problem (6.5) by

$$f(M) \triangleq \mathbf{Tr}\left(Y^T \left(\mathbf{I} + \frac{1}{n\lambda} XMX^T\right)^{-1} Y\right). \tag{6.6}$$

Its gradient at a given $M \in \mathcal{M}$ is

$$\nabla f(M) = -\frac{1}{2n^2\lambda} X^T \left(\mathbf{I} + \frac{1}{n\lambda} XMX^T\right)^{-1} YY^T \left(\mathbf{I} + \frac{1}{n\lambda} XMX^T\right)^{-1} X. \tag{6.7}$$

Observe that $-\nabla f(M)$ is a semi-definite positive matrix of squared root

$$U = \frac{1}{n\sqrt{2\lambda}} X^T \left(\mathbf{I} + \frac{1}{n\lambda} X M X^T \right)^{-1} Y \in \mathbb{R}^{d \times K}.$$

The linear minimization oracle to minimize $f(M)$ over the convex hull of the set of normalized equivalence matrices can then be computed as follows

$$\begin{aligned}
\underset{S \in \text{hull}(\mathcal{M})}{\text{argmin}} \langle S, \nabla f(M) \rangle &\overset{\vartheta_1}{=} \underset{S \in \mathcal{M}}{\text{argmin}} \, \mathbf{Tr}(S^T \nabla f(M)) \\
&= \underset{S \in \mathcal{M}}{\text{argmin}} - \mathbf{Tr}(S U U^T) \\
&= \underset{S \in \mathcal{M}}{\text{argmin}} \, \mathbf{Tr}((\mathbf{I} - S) U U^T)) \\
&\overset{\vartheta_2}{=} \underset{S \in \mathcal{M}}{\text{argmin}} \, \|U - SU\|_F^2. \qquad (6.8)
\end{aligned}$$

In ϑ_1, we used that $\mathcal{M}$ is a set of atoms, so its convex hull is a polytope and linear minimization on it is equivalent to linear minimization on its vertices, i.e., $\mathcal{M}$. In ϑ_2, we used that normalized equivalence matrices are orthogonal projectors, so $I - S$ is also an orthogonal projector. Now we observe that (6.8) is a k-means problem as presented in Section 5.2.2.

Therefore solving a k-means problem on the rows of the squared root matrix U of $-\nabla f(M)$ offers a solution to the linear minimization oracle. Crucially here if $K = 1$ (i.e. for regression or binary classification), U is a vector such that this reduces to a k-means problem in one dimension that can be solved exactly in polynomial time by dynamic programming [Bellman, 1973; Wang and Song, 2011]. Otherwise careful initialization as made in k-means++ [Arthur and Vassilvitskii, 2007] offers logarithmic approximations to the problem.

A conditional gradient method can then be applied in our setting, so that we can solve a relaxed version of (6.5). Once done, it remains to provide an approximate feasible solution for the original constraints. Two projections of the relaxed version are possible: either finding the closest normalized equivalence matrix in Frobenius norm or computing the point that minimizes the gradient of the relaxed solution, i.e. computing its linear minimization oracle. In practice we chose second solution as it provided better results. Notice that the k-means operation used for the linear minimization provides not only a normalized equivalence matrix but also a corresponding assignment matrix Z that leads to the optimal coefficients that minimize $L_{\text{square}}(ZV) + \lambda R_{\text{square}}(ZV)$ as

$$V(Z) = (Z^T X^T X Z + \lambda n Z^T Z)^\dagger Z^T X^T Y. \qquad (6.9)$$

To summarize, our convex relaxation for regression with grouped features is presented in Algorithm 6. We denote by $Z = \text{k-means}(U, Q)$ an assignment matrix solution of the k-means problem that cluster rows of U in Q groups.

Algorithm 6 Convex relaxation for regression with grouped features

Inputs: Data (X, Y), desired number of groups Q, target precision ε, regularization parameter $\lambda \geq 0$
Initialize $M_0 \in \mathcal{M}$
for $t = 0, \ldots$ **do**
Compute $-\nabla f(M_t)$ in (6.7) and its squared root U
Get linear minimization oracle by computing

$$\begin{aligned} Z_t &= \text{k-means}(U, Q) \\ S_t &= Z_t (Z_t^T Z_t)^\dagger Z_t^T \end{aligned}$$

if $\mathbf{Tr}\left((M_t - S_t)^T \nabla f(M_t)\right) \leq \varepsilon$ **then** Stop **end if**
Set $M_{t+1} = M_t + \frac{1}{t+2}(S_t - M_t)$
end for
$\hat{Z} = Z_t$
$\hat{V} = V(\hat{Z})$ in (6.9)
Output: $\hat{W} = \hat{Z}\hat{V}$

6.2.3 Computational complexity

We briefly examine the complexity of Algorithm 6. For regression the k-means operation has a complexity of $O(d^2Q)$ operations to get an exact solution by dynamic programming. For classification, k-means++ initialization costs $O(Q^2d)$ operations and standard alternating minimization approximates the k-means operation at a cost of $O(TQd)$, where T is the number of alternating steps, that is generally small.

The squared root computation is directly given by the computation of $\nabla f(M)$. However, this gradient itself requires the inversion of a matrix of size $d \times d$. This burdens its implementation for big data sets. Yet, a few iterations of this algorithm can give a good initialization for non-convex approaches of the problem.

6.3 Iterative Hard Clustering

Previous part gave an overview of the tightest convex relation in the simple case of a squared loss. Convexity ensures convergence of the Algorithm 6, at least for regression or binary classification. However, as noticed, this scheme may not scale for big data. Moreover it does not transpose to losses different than the squared one. We therefore propose to tackle directly the non-convex problem

$$\begin{array}{ll} \text{minimize} & L(W) + \lambda R(W) \\ \text{subject to} & W = ZV, \quad Z \in \{0,1\}^{d \times Q}, \quad Z\mathbf{1} = \mathbf{1} \end{array} \tag{P}$$

in variables $W \in \mathbb{R}^{d \times K}$, $V \in \mathbb{R}^{Q \times K}$ and Z, where $\lambda \geq 0$ is a regularization parameter. We use a projected gradient scheme that amounts to iteratively cluster features at each gradient step. This transposes the Iterative Hard Thresholding [Blumensath and

Davies, 2009] algorithm, studied in compressed sensing, to the problem of grouping features. It offers a scalable solution for various losses as shown in the numerical experiments. Here we detail its implementation and analyze its performance from a compressed sensing point of view in the next section.

6.3.1 Projected gradient descent

The algorithm relies on the fact that projecting a point W on the feasible set, that reads

$$\begin{array}{ll} \text{minimize} & \|W - ZV\|_F^2 \\ \text{subject to} & Z \in \{0,1\}^{d\times Q}, \quad Z\mathbf{1} = \mathbf{1} \end{array}$$

in variable $V \in \mathbb{R}^{Q\times K}$ and Z is a clustering problem

$$\text{minimize} \sum_{q=1}^{Q} \sum_{i\in g_q} \|w_i - v_q\|_2^2, \tag{6.10}$$

in variables $v_1, \ldots, v_Q \in \mathbb{R}^K$ that are the centroids of the clusters and $G = (g_1, \ldots, g_Q)$ a partition of $\{1, \ldots, d\}$. As noticed in previous section, this k-means problem can be solved approximately with k-means++ if $K > 1$ or exactly in polynomial time if $K = 1$. Given a matrix W, whose rows we want to cluster in Q groups, we denote by $[Z, V] = \text{k-means}(W, Q)$ respectively the assignment matrix and the matrix of centroids output by a clustering algorithm. A projected gradient scheme for problem (P) is described in Algorithm 7 and its implementations details are provided in next section.

Algorithm 7 Iterative Hard Clustering

Inputs: Data (X, Y), desired number of groups Q, regularization parameter $\lambda \geq 0$, step size γ_t
Initialize $W_0 \in \mathbb{R}^{d\times K}$
for t = 1,...,T **do**
  $W_{t+1/2} = W_t - \gamma_t(\nabla L(W_t) + \lambda \nabla R(W_t))$
  $[Z_{t+1}, V_{t+1}] = \text{k-means}(W_{t+1/2}, Q)$
  $W_{t+1} = Z_{t+1} V_{t+1}$
end for
Output: $\hat{W} = W_T$

6.3.2 Detailed implementation

In practice, we stop the algorithm when change in objective values of (P) are below some prescribed threshold ε. We use a backtracking line search on the stepsize γ_t that guarantees decreasing of the objective. At each iteration if

$$\bar{W}_{t+1} \quad = \text{k-means}\left(W_t - \gamma_t(\nabla L(W_t) + \lambda \nabla R(W_t)), Q\right)$$

decreases the objective value we keep it and we increase the stepsize by a constant factor $\gamma_{t+1} = \alpha\gamma_t$ with $\alpha > 1$. If $\bar{W}_{t+1}$ increases the objective value we decrease the stepsize by a constant factor $\gamma_t \leftarrow \beta\gamma_t$, with $\beta < 1$, compute new $\bar{W}_{t+1}$ and iterate this operation until $\bar{W}_{t+1}$ decreases the objective value or the stepsize reaches the stopping value ε used as a stopping criterion on the objective values. We observed better results with this line search than with constant stepsize, in particular when the number of samples is small.

Using this strategy, we observed convergence of the projected gradient algorithm in less than 100 iterations which makes it highly scalable. The complexity of its core operations amounts indeed to k-means operations whose complexities were given in Section 6.2.3.

6.4 Recovery performance of Iterative Hard Clustering

We now analyze convergence of the Iterative Hard Clustering scheme to retrieve the true regressor w_* in the regression problem. To this end we first detail the problem in terms of partitions.

6.4.1 Combinatorial penalty for grouping features

Several works developed tools to encode structural information in optimization problems such as sparsity inducing norms. Bach et al. [2013] show that these regularizers can generally be seen as convex extensions of combinatorial functions. A given vector w defines indeed by its support $\mathrm{Supp}(w) = \{i \in \{1, \dots, d\}, w_i \neq 0\}$ a set that can be constrained to satisfy some combinatorial properties by using submodular functions. For example, classical sparsity enforces the cardinality of this support to be small in order to select a few parameters. On a regression problem (6.1), this reads

$$\begin{array}{ll} \text{minimize} & L(w) + \lambda R(w) \\ \text{subject to} & \mathbf{Card}(\mathrm{Supp}(w)) \leq s, \end{array}$$

in variable $w \in \mathbb{R}^d$, where s is the desired sparsity. In our context, we do not use the set defined by the support of a vector, but the partition given by its level sets, that read

$$\mathrm{Part}(w) = \{g \subset \{1, \dots, d\} : (i, j) \in g \times g, \text{ iff } w_i = w_j\}.$$

Denoting $\mathbf{Card}(G)$ the number of the (non-empty) groups of a partition G, linear regression enforcing Q group of features (6.2) then reads

$$\begin{array}{ll} \text{minimize} & L(w) + \lambda R(w) \\ \text{subject to} & \mathbf{Card}(\mathrm{Part}(w)) \leq Q, \end{array} \tag{6.11}$$

in variable $w \in \mathbb{R}^d$. Assignment matrices introduced in Section 6.1 encode then the possible partitions into at most Q groups. We detail the geometrical interpretations

of assignment and normalized equivalence matrices in Appendix 6.A. Identifying the underlying combinatorial constraint helps then to describe the geometry of the feasible set. It can also be used to derive a norm as discussed in Appendix 6.B.

6.4.2 Geometry induced by partitions

Denote $\mathcal{P}$ the set of partitions of $\{1,\dots,d\}$ whose definition is recalled below.

Definition 6.4.1. ***Partitions*** *A collection G of subsets of $\{1,\dots,d\}$ is a partition of $\{1,\dots,d\}$ if for any $g,g' \in G\times G$, $g \neq g'$ implies $g\cap g' = \emptyset$ and if $\bigcup_{g\in G} g = \{1,\dots,d\}$.*

Denote from now on $\mathcal{P}$ the set of partitions of $\{1,\dots,d\}$. Pair of partitions can then be compared as follows.

Definition 6.4.2. ***Sup- and sub-partitions*** *Let $G, G' \in \mathcal{P}$ be two partitions. G is a* sup-partition *of G' (or G' is a* sub-partition *of G), denoted*

$$G \succeq G',$$

if for any $g' \in G'$ there exists $g \in G$ such that $g' \subset g$, or equivalently if any $g \in G$ is a union of groups g' of G'.

Relation $\succeq$ is transitive, reflexive and anti-symmetric, such that it is a partial order on the set of partitions. Notice that the number of (non-empty) groups of partitions, $\mathbf{Card}(\cdot)$, decreases with the partial order $\succeq$.

Following proposition highlights the geometry induced by a single partition of the features.

Proposition 6.4.3. *Any partition $G \in \mathcal{P}$ defines a linear subspace*

$$E_G = \{w \in \mathbb{R}^d : \mathrm{Part}(w) \succeq G\} \tag{6.12}$$

of vectors w whose level sets can be partitioned by the groups of G. For any partitions $G, G' \in \mathcal{P}$, if $G \succeq G'$ then $E_G \subset E_{G'}$.

Proof. Given a partition $G \in \mathcal{P}$ and a vector $w \in \mathbb{R}^d$, G is a sub-partition of $\mathrm{Part}(w)$, i.e. $\mathrm{Part}(w) \succeq G$, if and only if the groups of G are subsets of equal coefficients of w, or equivalently if level sets of w can be partitioned by groups of G. Now, if, for some $w_1, w_2 \in \mathbb{R}^d$, groups of G are subsets of equal coefficients of both w_1 and w_2, they will also be subset of equal coefficients of any linear combination of w_1, w_2. Therefore E_G is a linear subspace. Second statement follows from the transitivity of $\succeq$. ∎

Since $w \in E_{\mathrm{Part}(w)}$, the feasible set for the regression problem (6.11) enforcing Q groups of features is then a union of subspaces:

$$\{w \in \mathbb{R}^d : \mathbf{Card}(\mathrm{Part}(w)) \leq Q\} = \bigcup_{G\in\mathcal{P}\,:\,\mathbf{Card}(G)\leq Q} E_G = \bigcup_{G\in\mathcal{P}\,:\,\mathbf{Card}(G)=Q} E_G.$$

Second equality comes from the fact that if a partition $G \in \mathcal{P}$ has strictly less than Q groups, i.e., $\mathbf{Card}(G) < Q$, some of its groups can always be split to form a new partition G' such that $G \succeq G'$, $\mathbf{Card}(G') = Q$ and $E_G \subset E_{G'}$. Therefore it is sufficient to consider subspaces generated by partitions into exactly Q groups, whose set is denoted $\mathcal{P}_Q = \{G \in \mathcal{P} : \mathbf{Card}(G) = Q\}$.

6.4.3 Convergence analysis of Iterative Hard Clustering

We now analyze the convergence of the projected gradient algorithm applied to a regression problem enforcing Q groups of features. We use a squared loss and no regularization. Therefore our problem reads

$$\begin{array}{ll} \text{minimize} & \frac{1}{2n}\|Xw - y\|_2^2 \\ \text{subject to} & \mathbf{Card}(\text{Part}(w)) \leq Q \end{array} \tag{6.13}$$

in $w \in \mathbb{R}^d$, where $X = (x_1, \ldots, x_n)^T \in \mathbb{R}^{n \times d}$ is the matrix of data points and $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$ is the vector of observations. For the analysis, we use a constant step size $\gamma_t = 1$ and initialize the algorithm with $w_0 = 0$. We assume that the observations y are generated by a linear model whose coefficients w_* satisfy the constraints above, up to additive noise, that is

$$y = Xw_* + \eta,$$

where $\eta \sim \mathcal{N}(0, \sigma^2)$ and $\mathbf{Card}(\text{Part}(w_*)) \leq Q$. Hence we analyze the performance of the algorithm to recover w_* and the partition $\text{Part}(w_*)$ of the features.

If it were not the constraints, a gradient descent applied to (6.13) would act as a fixed point algorithm whose contraction factor depends on the singular values of the Hessian X^TX of the problem. Here we will show that the projected gradient scheme exhibits the same behavior, except that the contraction factor will depend on restricted singular values on small subspaces defined by partitions. These subspaces belong to the following collections

$$\begin{aligned} \mathcal{E}_1 &= \{E_G : G \in \mathcal{P}_Q\} \\ \mathcal{E}_2 &= \{E_{G_1} + E_{G_2} : G_1, G_2 \in \mathcal{P}_Q\} \\ \mathcal{E}_3 &= \{E_{G_1} + E_{G_2} + E_{G_3} : G_1, G_2, G_3 \in \mathcal{P}_Q\}. \end{aligned} \tag{6.14}$$

Throughout the rest of the section, for a given subspace E of $\mathbb{R}^d$, we denote P_E the orthogonal projector on E, U_E any orthonormal basis of E and for a given matrix $X \in \mathbb{R}^{n \times d}$ we denote $\sigma_{\min}(XU_E/\sqrt{n})$ and $\sigma_{\max}(XU_E/\sqrt{n})$ respectively the smallest and largest singular values of $XU_E/\sqrt{n}$, i.e. the smallest and largest restricted singular values of $X/\sqrt{n}$ on E.

The next proposition adapts the proof of Iterative Hard Thresholding in our context using that feasible set is a union of subspaces,

Proposition 6.4.4. *Iterative Hard Clustering Algorithm 7 with constant step size $\gamma_t = 1$ and initialization $w_0 = 0$, applied to (6.13) outputs iterates w_t that converge*

to the original w_* as

$$\|w_* - w_t\|_2 \leq \rho^t \|w_*\|_2 + \frac{1-\rho^t}{1-\rho}\nu\|\eta\|_2,$$

where

$$\begin{aligned}\rho &= 6 \max_{E\in\mathcal{E}_2} \max(\delta_E, \delta_E^3),\\ \nu &= 2/\sqrt{n} \max_{E\in\mathcal{E}_3} \sigma_{\max}(XU_E/\sqrt{n})\end{aligned}$$

and for any subspace E of $\mathbb{R}^d$, δ_E is the smallest non-negative constant that satisfies

$$1 - \delta_E \leq \sigma_{\min}\left(XU_E/\sqrt{n}\right) \leq \sigma_{\max}\left(XU_E/\sqrt{n}\right) \leq 1 + \delta_E.$$

Proof. To describe Algorithm 7, we define for $t \geq 0$,

$$\begin{aligned}w_{t+1/2} &= w_t - \gamma_t \nabla L(w_t) = w_t - \frac{1}{n}X^T X(w_t - w_*) + \frac{1}{n}X^T\eta\\ w_{t+1} &= \underset{w\in\mathbb{R}^d\,:\,\mathbf{Card}(\mathrm{Part}(w))\leq Q}{\operatorname{argmin}} \|w - w_{t+1/2}\|_2^2,\end{aligned}$$

where w_{t+1} is given exactly by the solution of a k-means problem in one dimension. The analysis of convergence relies on the characterization of the subspaces that contain w_*, w_t and w_{t+1}. We define therefore

$$\begin{aligned}E_{t,*} &= E_{\mathrm{Part}(w_t)} + E_{\mathrm{Part}(w_*)}\\ E_{t+1,*} &= E_{\mathrm{Part}(w_{t+1})} + E_{\mathrm{Part}(w_*)}\\ E_{t,t+1,*} &= E_{\mathrm{Part}(w_t)} + E_{\mathrm{Part}(w_{t+1})} + E_{\mathrm{Part}(w_*)},\end{aligned}$$

and the orthogonal projections on these set respectively $P_{t,*}, P_{t+1,*}, P_{t,t+1,*}$. Bound on the error can then be computed as follows:

$$\begin{aligned}\|w_* - w_{t+1}\|_2 &= \|P_{t+1,*}(w_* - w_{t+1})\|_2\\ &\leq \|P_{t+1,*}(w_* - w_{t+1/2})\|_2 + \|P_{t+1,*}(w_{t+1/2} - w_{t+1})\|_2.\end{aligned} \tag{6.15}$$

In the second term, as $\mathbf{Card}(\mathrm{Part}(w_*)) \leq Q$ and $w_{t+1} = \underset{w\in\mathbb{R}^d\,:\,\mathbf{Card}(\mathrm{Part}(w))\leq Q}{\operatorname{argmin}} \|w - w_{t+1/2}\|_2^2$, we have

$$\|w_{t+1} - w_{t+1/2}\|_2^2 \leq \|w_* - w_{t+1/2}\|_2^2$$

which is equivalent to

$$\|P_{t+1,*}(w_{t+1} - w_{t+1/2})\|_2^2 + \|(I - P_{t+1,*})w_{t+1/2}\|_2^2 \leq \|P_{t+1,*}(w_* - w_{t+1/2})\|_2^2 + \|(I - P_{t+1,*})w_{t+1/2}\|_2^2$$

and this last statement implies

$$\|P_{t+1,*}(w_{t+1} - w_{t+1/2})\|_2 \leq \|P_{t+1,*}(w_* - w_{t+1/2})\|_2.$$

This means that we get from (6.15)

$$\begin{aligned}
\|w_* - w_{t+1}\|_2 &\leq 2\|P_{t+1,*}(w_* - w_{t+1/2})\|_2 \\
&= 2\|P_{t+1,*}(w_* - w_t - \frac{1}{n}X^TX(w_* - w_t) - \frac{1}{n}X^T\eta)\|_2 \\
&\leq 2\|P_{t+1,*}(I - \frac{1}{n}X^TX)(w_* - w_t)\|_2 + \frac{2}{n}\|P_{t+1,*}(X^T\eta)\|_2 \\
&= 2\|P_{t+1,*}(I - \frac{1}{n}X^TX)P_{t,*}(w_* - w_t)\|_2 + \frac{2}{n}\|P_{t+1,*}(X^T\eta)\|_2 \\
&\leq 2\|P_{t+1,*}(I - \frac{1}{n}X^TX)P_{t,*}\|_2\|w_* - w_t\|_2 + \frac{2}{n}\|P_{t+1,*}X^T\|_2\|\eta\|_2.
\end{aligned}$$

Now, assuming

$$2\|P_{t+1,*}(I - \frac{1}{n}X^TX)P_{t,*}\|_2 \leq \rho \tag{6.16}$$

$$\frac{2}{n}\|P_{t+1,*}X^T\|_2 \leq \nu \tag{6.17}$$

and developing the latter inequality over t, using that $w_0 = 0$, we get

$$\|w_* - w_t\|_2 \leq \rho^t\|w_*\|_2 + \frac{1-\rho^t}{1-\rho}\nu\|\eta\|_2.$$

Bounds ρ and ν can then be given by restricted singular values of X. For ν in (6.17), we have

$$\|P_{t+1,*}X^T\|_2 = \|XP_{t+1,*}\|_2 \overset{\vartheta}{\leq} \max_{E\in\mathcal{E}_2}\|XP_E\|_2 = \max_{E\in\mathcal{E}_2}\sigma_{\max}(XU_E).$$

For ϑ, as noticed in previous section, if for example $\mathbf{Card}(\mathrm{Part}(w_*)) < Q$, there always exists $G \in \mathcal{P}$ such that $\mathrm{Part}(w_*) \succeq G$, $\mathbf{Card}(G) = Q$ and so $E_{\mathrm{Part}(w_*)} \subset E_G$. Therefore there exists $F_{t+1,*}$ that contain $E_{t+1,*}$ and belong to $\mathcal{E}_2$, such that we can restrict our attention to restricted singular values on subspaces in $\mathcal{E}_2$ (defined from partitions in exactly Q groups).

For ρ in (6.16), we have

$$\begin{aligned}
\|P_{t+1,*}(I - X^TX)P_{t,*}\|_2 &\overset{\vartheta_1}{\leq} \|P_{t,t+1,*}(I - \frac{1}{n}X^TX)P_{t,t+1,*}\|_2 \\
&\overset{\vartheta_2}{\leq} \max_{E\in\mathcal{E}_3}\|P_E(I - \frac{1}{n}X^TX)P_E\|_2 \\
&= \max_{E\in\mathcal{E}_3}\|U_E(I - \frac{1}{n}U_E^TX^TXU_E)U_E^T\|_2 \\
&= \max_{E\in\mathcal{E}_3}\|I - \frac{1}{n}U_E^TX^TXU_E\|_2,
\end{aligned}$$

where for a subspace E, U_E denotes any orthonormal basis of it. In ϑ_1 we used that $E_{t,t+1,*}$ contain $E_{t,*}$ and $E_{t+1,*}$. In ϑ_2 we use the same argument as for ν to restrict

our attention to subspaces defined by partitions into exactly Q groups. Finally, for a subspace E if $\delta_E \geq 0$ satisfies

$$1 - \delta_E \leq \sigma_{\min}\left(XU_E/\sqrt{n}\right) \leq \sigma_{\max}\left(XU_E/\sqrt{n}\right) \leq 1 + \delta_E,$$

then [Vershynin, 2010, Lemma 5.38] shows

$$\|I - \frac{1}{n}U_E^T X^T X U_E\|_2 \leq 3\max\{\delta_E, \delta_E^2\},$$

which concludes the proof by taking the maximum of δ_E over $\mathcal{E}_3$. ∎

If the contraction factor is sufficient, convergence of the projected gradient scheme to the original vector is ensured up to a constant error of the order of the noise as the classical IHT algorithm does for sparse signals [Blumensath and Davies, 2009].

6.4.4 Recovery performance on random instances

We observe now that for isotropic independent sub-Gaussian data x_i the restricted singular values introduced in Proposition 6.4.4 depend on the number of subspaces that define partitions and their dimension. This proposition reformulates results of Vershynin [2010, Theorems 5.39, 5.65] in our context.

Proposition 6.4.5. *Let $\mathcal{E}$ be a collection of subspaces of $\mathbb{R}^d$ of dimension at most D and denote N their number. If the samples are n isotropic independent sub-gaussian random variables forming a design matrix $X = (x_1, \ldots, x_n)^T \in \mathbb{R}^{n\times d}$, then for any $E \in \mathcal{E}$,*

$$1 - \delta - \varepsilon \leq \sigma_{\min}\left(\frac{XU_E}{\sqrt{n}}\right) \leq \sigma_{\max}\left(\frac{XU_E}{\sqrt{n}}\right) \leq 1 + \delta + \varepsilon,$$

with probability larger than $1 - \exp(-c\varepsilon^2 n)$, where $\delta = C_0\sqrt{\frac{D}{n}} + \sqrt{\frac{\log(N)}{cn}}$ and C_0, c depend only on the sub-gaussian norm of the x_i.

Proof. Let $E \in \mathcal{E}$, denote U_E one of its orthonormal basis and $D_E = \dim(E) \leq D$ its dimension. The rows of XU_E are orthogonal projections of the rows of X onto E, so they are still independent sub-gaussian isotropic random vectors. We can therefore apply [Vershynin, 2010, Theorem 5.39] on $XU_E \in \mathbb{R}^{n\times D_E}$. Hence, for any $s \geq 0$, with probability at least $1 - 2\exp(-cs^2)$, the smallest and largest singular values of $XU_E/\sqrt{n}$ are bounded as

$$1 - C_0\sqrt{\frac{Q}{n}} - \frac{s}{\sqrt{n}} \leq \sigma_{\min}\left(\frac{XU_E}{\sqrt{n}}\right) \leq \sigma_{\max}\left(\frac{XU_E}{\sqrt{n}}\right) \leq 1 + C_0\sqrt{\frac{Q}{n}} + \frac{s}{\sqrt{n}}, \quad (6.18)$$

where c and C_0 depend only on the sub-gaussian norm of the x_i. Now, by taking the union bound, (6.18) holds for any $G \in \mathcal{P}_Q$ with probability $1 - 2N\exp(-cs^2)$.

Taking $s = \sqrt{\frac{\log(N)}{c}} + \varepsilon\sqrt{n}$, we get for all $G \in \mathcal{P}_Q$,

$$1 - \delta - \varepsilon \leq \sigma_{\min}\left(\frac{XU_E}{\sqrt{n}}\right) \leq \sigma_{\max}\left(\frac{XU_E}{\sqrt{n}}\right) \leq 1 + \delta + \varepsilon,$$

with probability at least $1 - 2\exp(-c\varepsilon^2 n)$, where $\delta = C_0\sqrt{\frac{Q}{n}} + \sqrt{\frac{\log(N)}{cn}}$. ∎

To ensure approximate recovery of the projected gradient scheme in Proposition 6.4.4, one needs to control restricted singular values of X on subspaces in $\mathcal{E}_3$ in order to ensure that the contraction factor ρ is strictly less than one. Precisely, we need to ensure for that for any $E \in \mathcal{E}_3$, there exists $0 \leq \delta < 1/6$ such that

$$1 - \delta \leq \sigma_{\min}\left(\frac{XU_E}{\sqrt{n}}\right) \leq \sigma_{\max}\left(\frac{XU_E}{\sqrt{n}}\right) \leq 1 + \delta.$$

Denoting D_3 and N_3 respectively the largest dimension of the subspaces in $\mathcal{E}_3$ and the number of these subspaces, last proposition shows that when observations x_i are isotropic independent sub-gaussian, their number n must therefore satisfy

$$C_0\sqrt{\frac{D_3}{n}} < \frac{1}{6} \quad \text{and} \quad \sqrt{\frac{\log(N_3)}{cn}} < \frac{1}{6}$$

which is roughly

$$n = \Omega(D_3) \quad \text{and} \quad n = \Omega(\log(N_3)) \tag{6.19}$$

The first condition in (6.19) means that subspaces must be low-dimensional, in our case $D_3 = 3Q$, and we naturally want the number of groups to be small. The second condition in (6.19) means that the structure (partitioning here) is restrictive enough, i.e., that the number of possible configurations, N_3, is small enough.

To compute N_3, denote N_Q the number of partitions in exactly Q groups such that $N_3 = \binom{N_Q}{3}$. The number of partitions into Q groups is then given by the the Stirling number of second kind $N_Q = \left\{ {d \atop Q} \right\}$, that can be bounded as

$$Q^{d-Q} \leq \left\{ {d \atop Q} \right\} \leq \frac{1}{2}(ed/Q)^Q Q^{d-Q}. \tag{6.20}$$

Using standard bounds on the binomial coefficients this means

$$N_3 \geq \left(\frac{N_Q}{3}\right)^3 \geq \frac{Q^{3d-3Q}}{27}.$$

Therefore although the intrinsic dimension of our variables is of order $3Q$, the number of subspaces N_3 is such that we need roughly $n \geq 3d\log(Q)$ observations, i.e., approximately as many samples as features, so the grouping structure is not specific enough to reduce the number of samples required by a projected gradient scheme to converge. On the other hand, given this many samples, the algorithm provably

converges to the original w^*, which helps interpretation.

As a comparison, classical sparse recovery problems have the same structure [Rao et al., 2012], as s-sparse vectors for instance can be described as $\{w = Zv, Z \in \{0,1\}^{d\times s}, Z^T 1 = 1\}$ and so are part of a "union of subspaces". However in the case of sparse vectors the number of subspaces grows as d^s which means recovery requires much less samples than features.

6.5 Sparse and grouped linear models

Projected gradient schemes are simple but scalable algorithms to tackle constrained structures of linear models. It has been developed for sparsity through the Iterative Hard Thresholding algorithm [Blumensath and Davies, 2009], we presented its version to group features in Section 6.3, we now extend it to both select s features and group them in Q groups. Using the notations introduced in Section 6.4.1, a regression problem that enforces predictors to have at most s non-zeros coefficients clustered in at most Q groups reads

$$\begin{array}{ll} \text{minimize} & L(w) + \lambda R(w) \\ \text{subject to} & \mathbf{Card}(\mathrm{Supp}(w)) \leq s, \quad \mathbf{Card}(\mathrm{Part}(w)) \leq Q + 1 \end{array} \tag{6.21}$$

in variable $w \in \mathbb{R}^d$, where L and R are respectively the loss and the regularizer of the prediction problem as introduced in Section 6.1 and $\lambda \geq 0$ is a regularization parameter. Naturally we take $Q \leq s$ as one cannot cluster s features in more than s groups.

A projected gradient for this problem requires essentially an efficient algorithm for the projection step. To this end, we develop a new dynamic program to get the projection on s-sparse vectors whose non-zero coefficients form Q groups of equal coefficients. Analysis of the recovery performance of this scheme will then follow from the previous study for grouped vectors.

6.5.1 Projection on s-sparse Q-grouped vectors

Given two projectors on different subspaces, the projection on their intersection is not simply given by the composition of the projectors. Similarly here one easily shows that projection on sparse and clustered vectors is not given by a thresholding step followed by a clustering one, finer analysis is necessary as detailed below.

Formulation of the problem

A feasible point $w \in \mathbb{R}^d$ for problem (6.21) is described by the partition of its coordinates $G = \{g_0, \ldots, g_{Q_G}\}$ in groups of equal coefficients, where g_0 is the group of zero coefficients, and $v_1, \ldots, v_{Q_G}$ the possible values of the non-zero coefficients. $Q_G = \mathbf{Card}(G) - 1 \geq 0$ denotes here the number of (non-empty) groups of a partition $G \in \mathcal{P}$.

Let us fix a point $x \in \mathbb{R}^d$, its distance to a feasible point w reads

$$\|x - w\|_2^2 = \sum_{i \in g_0} x_i^2 + \sum_{q=1}^{Q_G} \sum_{i \in g_q} (x_i - v_q)^2, \tag{6.22}$$

for given $G = \{g_0, \dots g_{Q_G}\} \in \mathcal{P}$ and $v \in \mathbb{R}^{Q_G}$. For a fixed partition $G \in \mathcal{P}$, hence a fixed subspace, minimization in v gives the barycenters of the groups $g_1, \dots, g_{Q_G}$ of non-zero coefficients denoted

$$\mu_q = \frac{1}{s_q} \sum_{i \in g_q} x_i \quad \text{for } q = 1, \dots Q_G,$$

where $s_q = \mathbf{Card}(g_q)$ is the size of the q^{th} group. Inserting them in (6.22), the distance to a subspace of sparse grouped coefficients defined by a partition $G \in \mathcal{P}$ can be developed as

$$\sum_{i \in g_0} x_i^2 + \sum_{q=1}^{Q_G} \sum_{i \in g_q} (x_i^2 + \mu_q^2 - 2 v_q x_i) = \sum_{i=1}^{d} x_i^2 - \sum_{q=1}^{Q_G} s_q \mu_q^2.$$

Projection on the feasible set of (6.21), that minimizes the above distance for all possible partitions in Q groups of s non-zeros coefficients, amounts then to solve

$$\begin{array}{ll} \text{maximize} & \sum_{q=1}^{Q_G} s_q \mu_q^2 \\ \text{subject to} & \mathbf{Card}\left(\bigcup_{q=1}^{Q_G} g_q\right) \leq s, \quad 0 \leq Q_G \leq Q, \end{array} \tag{6.23}$$

in the partition $G = \{g_0, \dots, g_{Q_G}\} \in \mathcal{P}$, where $\mu_q = \frac{1}{s_q} \sum_{i \in g_q} x_i$ and $Q_G = \mathbf{Card}(G) - 1$.

This problem amounts to select a number $s' \leq s$ of features and cluster them in a number $Q' \leq Q$ groups whose barycenters have maximal magnitude for the objective in (6.23). The objective can then be split into positive and negative barycenters to treat each resulting problem independently and then find the best balance between both parts.

Dynamic programing

To solve problem (6.23), observe first that the objective is clearly increasing with the number of groups, as it allows more degrees of freedom to approximate x. Furthermore if the number s' of selected features is fixed, the number of groups cannot exceed it, i.e. $Q' \leq s'$, and it can therefore be set at $\min(s', Q)$.

A solution of (6.23) that selects $s' \leq s$ features is then composed of a partition of j points into q groups that define positive barycenters, and a partition of the $s' - j$ remaining points into $\min(s', Q) - q$ groups that define negative centers. We therefore tackle (6.23) by searching for the best parameters s', j, q that balance optimally the objective into positive and negative barycenters.

To this end, we define $f_+(j,q)$ the optimal value of (6.23) when picking j points clustered in q groups of positive barycenters, *i.e.* the solution of the problem

$$\begin{array}{ll} \text{maximize} & \sum_{p=1}^{q} s_p \mu_p^2 \\ \text{subject to} & \mu_p = \frac{1}{s_p} \sum_{i \in g_p} x_i > 0 \\ & \mathbf{Card}\left(\bigcup_{p=1}^{q} g_p\right) = j, \end{array} \qquad (P_+(j,q))$$

in disjoint groups $g_1, \dots, g_q \subset \{1, \dots, d\}$. This problem is not always feasible, as it may not be possible to find q clusters of positive barycenters with j points. In that case we denote its solution $f_+(j,q) = +\infty$. We define similarly $f_-(j,q)$ the optimal value of (6.23) when picking j points clustered in q groups forming only negative barycenters. The best balance between the two, which solves (6.23), is then given by solving:

$$\begin{array}{ll} \text{maximize} & f_+(j,q) + f_-(s'-j, Q'-q) \\ \text{subject to} & 0 \le j \le s',\ 0 \le q \le Q', \\ & 0 \le s' \le s,\ Q' = \min(s', Q), \end{array} \qquad (6.24)$$

in variables s', j and q.

It remains to compute f_+ and f_- efficiently. We present our approach for f_+ that transposes to f_-. Let $S_+ \subset \{1, \dots, d\}$ be the optimal subset of indexes taken for $(P_+(j,q))$ and $i \in S_+$. If there exists $j \in \{1, \dots, d\} \setminus S_+$ such that $x_j \ge x_i$, then swapping j and i would increase the magnitude of the barycenter of the group that i belongs to and so the objective. Therefore $(P_+(j,q))$ amounts to a partitioning problem on the j largest values of x. From now on, assume coefficients of x to be in decreasing order $x_1 \ge \dots \ge x_d$. For $(P_+(j,q))$ a feasible problem, denote $g_1, \dots, g_q$ the optimal partition of $\{1, \dots, j\}$ whose corresponding barycenters are in decreasing order. Let i be the index of the largest coefficient of x in g_q, then necessarily $g_1, \dots, g_{q-1}$ is optimal to solve $(P_+(i-1, q-1))$. f_+ can then be computed recursively as

$$f_+(j,q) = \max_{\substack{q \le i \le j \\ \mu(x_i, \dots, x_j) > 0}} f_+(i-1, q-1) + (j-i+1)\mu(x_i, \dots, x_j)^2, \qquad (6.25)$$

where $\mu(x_i, \dots, x_j) = \frac{1}{j-i+1} \sum_{l=i}^{j} x_l$ can be computed in constant time using that

$$\mu(x_i, \dots, x_j) = \frac{x_i + (j-i)\mu(x_{i+1}, \dots, x_j)}{j-i+1}.$$

By convention $f_+(j,q) = -\infty$ if is not possible to find q clusters of positive barycenters with j points such that $(P_+(j,q))$ is not feasible. Values of f_+ are stored to compute (6.24). Two auxiliary variables I_+ and v_+ store respectively the indexes of the largest value of x in group g_q and the barycenter of the group g_q. The same dynamic program can be used to compute f_-, I_- and v_-, defined similarly as I_+ and v_+, by reversing the order of the values of x. A grid search on $f(j,q,s') = f_+(j,q) + f_-(s'-j, Q'-q)$, with $Q' = \min(s', Q)$, gives the optimal balance between positive and negative barycenters. A backtrack on I_- and I_+ finally gives the best partition and the projection with the associated barycenters given in v_- and v_+.

f_+ is initialized as a grid of $k+1$ and $Q+1$ columns such that $f_+(0,q) = 0$ for any q, $f_+(j,0) = 0$ and $f_+(j,1) = j\mu(x_1,\dots,x_j)^2$ for any $j \geq 1$. I_+ and v_+ are initialized by $I_+(j,1) = 1$ and $\mu_+(j,1) = \mu(x_1,\dots,x_j)$.

Each dynamic program needs only to build the best partitions for the s smallest or largest partitions so they cost $O(s^2Q)$ elementary operations. The grid search and the backtrack cost respectively $O(s^2Q)$ and $O(Q)$ elementary operations. Overall, the complexity of the projection does not exceed $O(s^2Q)$.

6.5.2 Recovery performance

Analysis of recovery performance of the projected gradient for sparse clustered vectors follows the one provided in Section 6.4. Our problem is to recover an original vector w_* such that $\mathbf{Card}(\mathrm{Supp}(w_*)) \leq s$ and $\mathbf{Card}(\mathrm{Part}(w_*)) \leq Q+1$ that generates n noisy observations y_i from data points x_i as

$$y = Xw_* + \eta,$$

where $\eta \sim \mathcal{N}(0,\sigma^2)$, where $X = (x_1,\dots,x_n)^T \in \mathbb{R}^{n\times d}$ is the matrix of data points and $y = (y_1,\dots,y_n) \in \mathbb{R}^n$ is the vector of observations. To this end we attempt to solve a regression problem enforcing Q groups of s features with a squared loss and no regularization, which reads

$$\begin{array}{ll} \text{minimize} & \frac{1}{2n}\|Xw - y\|_2^2 \\ \text{subject to} & \mathbf{Card}(\mathrm{Supp}(w)) \leq s, \quad \mathbf{Card}(\mathrm{Part}(w)) \leq Q+1. \end{array} \tag{6.26}$$

As in Section 6.4, we use a projected gradient scheme with constant step size $\gamma_t = 1$ and initialized at $w_0 = 0$, the only difference is the projection step that is given here by the dynamic program presented in last section.

First we detail the geometry of the feasible set of (6.21). A given subset $S \subset \{1,\dots,d\}$ defines a linear subspace

$$E_S = \{w \in \mathbb{R}^d : \mathrm{Supp}(w) \subset S\}.$$

By combining a subset $S \in \{1,\dots,d\}$ with a partition $G \in \mathcal{P}$ we get a linear subspace

$$E_{S,G} = \{w \in \mathbb{R}^d : \mathrm{Supp}(w) \subset S,\ \mathrm{Part}(w) \succeq G\} = E_S \cap E_G.$$

Vectors in $E_{S,G}$ have at most $\mathbf{Card}(G) - 1$ different non-zero coefficients such that $\dim(E_{S,G}) = \mathbf{Card}(G) - 1$. The feasible set of (6.21) is then a union of subspaces,

$$\{w \in \mathbb{R}^d : \mathbf{Card}(\mathrm{Supp}(w)) \leq s,\ \mathbf{Card}(\mathrm{Part}(w)) \leq Q+1\} = \bigcup_{\substack{S\in\{1,\dots,d\} : \mathbf{Card}(S)\leq s \\ G\in\mathcal{P} : \mathbf{Card}(G)\leq Q+1}} E_{S,G}.$$

Analysis of convergence made in Proposition 6.4.4 for the clustered case relies only on the fact that the feasible set is a union of subspaces and that the projection on it can be computed exactly, so it applies also here. However the contraction factor

will now depend on restricted singular values of the data on a smaller collection of subspaces. Precisely, define

$$\begin{aligned}\tilde{\mathcal{E}} &= \{E_{S,G} : S \subset \{1,\dots,d\},\ G \in \mathcal{P},\ \mathbf{Card}(S) = s,\ \mathbf{Card}(G) = Q+1\}\\ \tilde{\mathcal{E}}_3 &= \{E_1 + E_2 + E_3 : E_1, E_2, E_3 \in \tilde{\mathcal{E}}\}.\end{aligned}$$

The contraction factor depends then on the restricted singular values of the matrix X on subspaces belonging to $\tilde{\mathcal{E}}_3$. Since $\dim(E_{S,G}) = \mathbf{Card}(G) - 1$, subspaces in $\tilde{\mathcal{E}}_3$ have a dimension at most $3Q$. Denoting N and N_3 the cardinality of respectively $\tilde{\mathcal{E}}$ and $\tilde{\mathcal{E}}_3$, we have $N_3 = \binom{N}{3}$. Subspaces of $\tilde{\mathcal{E}}$ are defined by selecting s features among d and partitioning these s features into Q groups so that their number is $N = \binom{d}{s}\left\{{s \atop Q}\right\}$. Using classical bounds on the binomial coefficient and (6.20), we can roughly bound N for $s \geq 3$, $Q \geq 3$ by

$$N \leq \left(\frac{ed}{s}\right)^s \frac{1}{2}\left(\frac{e}{Q}\right)^Q s^Q Q^{s-Q} \leq d^s s^Q Q^{s-Q}$$

and so

$$N_3 \leq \left(\frac{eN}{3}\right)^3 \leq N^3 \leq (d^s s^Q Q^{s-Q})^3$$

Propositions 6.4.4 and 6.4.5 adapted in this case thus predict that the number of observations must satisfy

$$n = O(s \log d + Q \log(s) + (s-Q)\log(Q))$$

for a projected gradient scheme to recover approximately w_*. It produces $Q+1$ cluster of features, one being a cluster of zero features, reducing dimensionality, while needing roughly as many samples as non-zero features.

6.6 Numerical experiments

We now test our methods, first on artificial datasets to check their robustness to noisy data, then on real data extracted from movie reviews.

6.6.1 Synthetic dataset

We test the robustness of our algorithms for an increasing number of training samples or level of noise in the labels. We generate a linear model in dimension $d = 100$ with a vector $w_* \in \mathbb{R}^d$ that has only $Q = 5$ different values uniformly distributed around 0. We sample n Gaussian random points x_i with noisy observations $y_i = w^T x_i + \eta$, where $\eta \sim \mathcal{N}(0, \sigma^2)$. We vary the number of samples n or the level of noise σ and measure $\|w_* - \hat{w}\|_2$, the l_2 norm of the difference between the true vector of weights w^* and the estimated ones $\hat{w}$.

In Table 6.1 and 6.2, we study the performance of our model with a squared loss and regularized by the Euclidean norm. We solve it with the conditional gradient

algorithm (CG),the Iterative Hard Clustering algorithm (IHC) (initialized with the solution of Least Square followed by k-means) and the conditional gradient followed by IHC (CGIHC). We compare the proposed algorithms to Least Squares regularized by the squared norm (LS), Least Squares regularized by the squared norm followed by K-means on the weights (using associated centroids as predictors) (LSK) and OSCAR [Bondell and Reich, 2008]. For OSCAR we used a submodular approach [Bach et al., 2012] to compute the corresponding proximal algorithm, which makes it scalable. "Oracle" refers to the Least Square solution given the true assignments of features and can be seen as the best achievable error rate. When varying the number of samples, noise on labels is set to $\sigma = 0.5$ and when varying level of noise σ number of samples is set to $n = 150$. Regularization parameters of the models were all cross-validated using a logarithmic grid. Results were averaged over 50 experiments and figures after the $\pm$ sign correspond to one standard deviation.

	$n = 50$	$n = 75$	$n = 100$	$n = 125$	$n = 150$
Oracle	0.16±0.06	0.14±0.04	0.10±0.04	0.10±0.04	0.09±0.03
LS	61.94±17.63	51.94±16.01	21.41±9.40	1.02±0.18	0.70±0.09
LSK	62.93±18.05	57.78±17.03	10.18±14.96	0.31±0.19	0.19±0.12
IHC	63.31±18.24	52.72±16.51	5.52±14.33	**0.14**±0.09	**0.09**±0.04
CG	61.81±17.78	52.59±16.58	17.24±13.87	1.20±1.38	1.05±1.37
CGIHC	62.29±18.15	**50.15**±17.43	**0.64**±2.03	0.15±0.19	0.17±0.53
OS	**61.54**±17.59	52.87±15.90	11.32±7.03	1.25±0.28	0.71±0.10

Table 6.1 – Measure of $\|w_* - \hat{w}\|_2$, the l_2 norm of the difference between the true vector of weights w^* and the estimated ones $\hat{w}$ along number of samples n.

	$\sigma = 0.05$	$\sigma = 0.1$	$\sigma = 0.5$	$\sigma = 1$
Oracle	0.86±0.27	1.72±0.54	8.62±2.70	17.19±5.43
LS	7.04±0.92	14.05±1.82	70.39±9.20	140.41±18.20
LSK	1.44±0.46	2.88±0.91	19.10±12.13	48.09±27.46
IHC	**0.87**±0.27	**1.74**±0.52	**9.11**±4.00	26.23±18.00
CG	23.91±36.51	122.31±145.77	105.45±136.79	155.98±177.69
CGIHC	1.52±3.13	140.83±710.32	17.34±53.31	**24.80**±16.32
OS	14.43±2.45	18.89±3.46	71.00±10.12	140.33±18.83

Table 6.2 – Measure of $\|w_* - \hat{w}\|_2$, the l_2 norm of the difference between the true vector of weights w^* and the estimated ones $\hat{w}$ along level of noise σ.

We observe that both IHC and CGIHC give significantly better results than other methods and even reach the performance of the Oracle for $n > d$ and for small σ, while for $n \leq d$ results are in the same range.

6.6.2 Predicting ratings from reviews using groups of words.

We perform "sentiment" analysis of newspaper movie reviews. We use the publicly available dataset introduced by Pang and Lee [2005] which contains movie reviews paired with star ratings. We treat it as a regression problem, taking responses for y in $(0, 1)$ and word frequencies as covariates. The corpus contains $n = 5006$ documents and we reduced the initial vocabulary to $d = 5623$ words by eliminating stop words, rare words and words with small TF-IDF mean on whole corpus. We evaluate our algorithms for regression with clustered features against standard regression approaches: Least-Squares (LS), Least-Squares followed by k-means on predictors (LSK), Oscar regularization (OS), LASSO and Iterative Hard Thresholding (IHT). We also tested our projected gradient with sparsity constraint, initialized by the solution of LSK (PGS) or by the solution of CG (CGPGS). Number of clusters, sparsity constraints and regularization parameters were 5-fold cross-validated using respectively grids going from 5 to 15, $d/2$ to $d/5$ and logarithmic grids. Cross validation and training were made on 80% on the dataset and tested on the remaining 20% it gave $Q = 15$ number of clusters and $d/2$ sparsity constraint for our algorithms. Results are reported in Table 6.3, figures after the $\pm$ sign correspond to one standard deviation when varying the training and test sets on 20 experiments.

All methods perform similarly except IHT and Lasso whose hypotheses do not seem appropriate for the problem. Our approaches have the benefit to reduce dimensionality from 5623 to 15 and provide meaningful cluster of words. The clusters with highest absolute weights are also the ones with smallest number of words, which confirms the intuition that only a few words are very discriminative. We illustrate this in Table 6.4, picking randomly words of the four clusters within which associated predictor weights v_q have largest magnitude.

LS	LSK	IHC	CG	CGIHC	OS
1.51±0.06	1.53±0.06	1.52±0.06	1.58±0.07	1.49±0.08	1.47±0.07
	IHCS	CGIHCS	IHT	LASSO	
	1.53±0.06	1.49±0.07	2.19±0.12	3.77±0.17	

Table 6.3 – 100 $\times$ mean square errors for predicting movie ratings associated with reviews.

6.7 Conclusion

We analyzed how prediction problems can be simplified by constraining groups of features, within each a common weight is assigned. Our formulation enables the development of a projected gradient scheme or a tight convex relaxation for squared loss. This approach can then be generalized for grouping samples or tasks in the following chapters. Our theoretical analysis of the projected gradient highlights that, even if dimension is reduced by grouping features, the treatment of this structure is

2 most negative clusters	bad, awful, worst, boring, ridiculous, watchable, suppose, disgusting,
2 most positive clusters	perfect,hilarious,fascinating,great wonderfully,perfectly,goodspirited, world, intelligent,wonderfully,unexpected,gem,recommendation, excellent,rare,unique,marvelous,good-spirited, mature,send,delightful,funniest

Table 6.4 – Clustering of words on movie reviews. We show clusters of words within which associated predictor weights have largest magnitude. First row presents ones associated to a negative coefficient and therefore bad feelings about movies, second row ones to a positive coefficient and good feelings about movies.

much more complex than simple sparsity as it requires as many samples as features to ensure efficiency of a projected gradient scheme. Yet, by developing a dynamic program for sparse and grouped vectors we finally obtain a fast and scalable algorithm that drastically reduces the dimensionality of the problem while enjoying properties of sparse problems.

Notice that the combinatorial penalty we use, i.e. the number of non-empty groups of a partition, is not a submodular function on the lattice that forms partitions. Using submodular functions of partitions may lead to efficient algorithmic solutions to group features as it has been done for classical sparsity [Bach et al., 2013].

Appendix

6.A Geometric interpretation of algebraic tools

In Proposition 6.4.3 we defined subspaces from partitions of$\{1, \dots, d\}$. Here we relate assignment and normalized equivalence matrices to these subspaces. First for a partition $G = (g_1, \dots, g_Q)$ into Q groups, $w \in E_G$ has at most Q different coefficients and can be encoded using assignment matrices as presented in Section 6.1. In other words, for an assignment matrix Z of G, one has

$$E_G = \{w = Zv, v \in \mathbb{R}^d\}$$

Columns of Z are orthogonal since since groups are disjoints and not null if G has no empty groups. In this case, Z is therefore an orthogonal basis of E_G. The normalized equivalence matrix of G, that reads $M = Z(Z^TZ)^{-1}Z^T$ is then the orthogonal projector on E_G.

As mentioned in Section 5.2, several assignment matrices can encode a partition, i.e. several binary matrices form a basis of a subspace E_G. However E_G and its orthogonal projector, i.e. the normalized equivalence matrix, are for their part uniquely defined by G.

To represent more generally partitions of $\{1, \dots, d\}$ in any number of groups one can use binary matrices $Z \in \{0,1\}^{d\times d}$ that satisfy $Z\mathbf{1} = \mathbf{1}$. Number of non-zero columns of such matrices are then the number of groups of the partition they represent. Once again partitions can be represented by several assignment matrices but are in bijection with the set of normalized equivalence matrices

$$\mathfrak{M} = \{M = Z(Z^TZ)^{\dagger}Z \ , \ Z \in \{0,1\}^{d\times d} \ , \ Z\mathbf{1} = \mathbf{1}\}. \tag{6.27}$$

Number of groups of a partition G is then equal to the rank of its normalized equivalence matrix (the dimension of E_G), i.e. $\mathbf{Card}(G) = \mathbf{Rank}(M) = \mathbf{Tr}(M)$, since M is a projector.

6.B Norm for grouping features

In this section, we seek to develop a norm that induce groups of features by regularization rather than enforcing it by constraints as in (6.2) or (6.4). In Section 6.4, we highlighted that our framework constraints number of level sets of the variables,

i.e. the function

$$\Omega(w) = \mathbf{Card}(\mathrm{Part}(w)).$$

Following Obozinski and Bach [2012] ,we investigate how this combinatorial function can be incorporated in standard Euclidean regularization by finding the tightest convex homogeneous envelope of

$$\Omega_2 = \frac{1}{2}\|w\|_2^2 + \frac{1}{2}\,\mathbf{Card}(\mathrm{Part}(w)).$$

Following proposition details its formulation

Proposition 6.B.1. *The tightest convex homogeneous envelope of*

$$\Omega_2(w) = \frac{1}{2}\|w\|_2^2 + \frac{1}{2}\,\mathbf{Card}(\mathrm{Part}(w))$$

is

$$\|w\|_{\Omega_2} = \inf_{\substack{(x_M)_{M\in\mathcal{M}} \\ x=\sum_{M\in\mathcal{M}} M x_M}} \sum_{M\in\mathcal{M}} \mathbf{Tr}(M)^{1/2}\|Mx_M\|_2,$$

where $\mathfrak{M}$ defined in (6.27) is the set of normalized equivalence matrices of partitions of $\{1,\dots,d\}$.

$\|w\|_{\Omega_2}$ is a norm, whose dual norm is

$$\|w\|_{\Omega_2}^* = \max_{M\in\mathcal{M}} \frac{\|Mx\|_2}{\mathbf{Tr}(M)^{1/2}}.$$

Proof. First we give an algebraic formulation of the combinatorial function Ω. Given a vector $w \in \mathbb{R}^d$, $\mathrm{Part}(w)$ is the largest partition (in terms of $\succeq$ presented in Definition 6.4.2) in groups of equal coefficients of w. It defines therefore the smallest subspace (see Proposition 6.4.3) on which w lies. $\mathbf{Card}(\mathrm{Part}(w))$ is then the dimension of the smallest subspace defined from partitions, on which w lies. Using normalized equivalence matrices that are orthogonal projections on these subspaces, as mentioned in Appendix 6.A, the combinatorial penalty Ω reads

$$\Omega(w) = \mathbf{Card}(\mathrm{Part}(w)) = \min_{\substack{M\in\mathfrak{M} \\ Mw=w}} \mathbf{Tr}(M).$$

Now, following [Obozinski and Bach, 2012], we begin by computing the homogenized version of Ω_2 defined as $h(w) = \inf_{\lambda>0} \frac{\Omega_2(\lambda w)}{\lambda}$, then we compute the Fenchel bi-conjugate of h. We have

$$\begin{aligned} h(w) &= \inf_{\lambda>0} \frac{1}{2}\|w\|_2^2\lambda + \frac{1}{2}\Omega(w)\lambda^{-1}. \\ &= \|w\|_2\Omega(w)^{1/2} \end{aligned}$$

Fenchel dual of h reads then

$$\begin{aligned}
h^*(x) &= \sup_{w\in\mathbb{R}^d} x^T w - \|w\|_2 \Omega(w)^{1/2} \\
&= \sup_{w\in\mathbb{R}^d} \max_{\substack{M\in\mathfrak{M}\\ Mw=w}} x^T w - \|w\|_2 \,\mathbf{Tr}(M)^{\frac{1}{2}} \\
&= \max_{M\in\mathfrak{M}} \sup_{\substack{w\in\mathbb{R}^d\\ Mw=w}} x^T w - \|w\|_2 \,\mathbf{Tr}(M)^{\frac{1}{2}} \\
&= \max_{M\in\mathfrak{M}} \begin{cases} 0 & \text{if} \quad \|Mx\|_2 \leq \mathbf{Tr}(M)^{1/2} \\ +\infty & \text{otherwise} \end{cases} \\
&= \begin{cases} 0 & \text{if} \quad \max_{M\in\mathcal{M}} \|Mx\|_2 \,\mathbf{Tr}(M)^{-1/2} \leq 1 \\ +\infty & \text{otherwise.} \end{cases}
\end{aligned}$$

Define

$$\|w\|^*_{\Omega_2} = \max_{M\in\mathfrak{M}} \|Mx\|_2 \,\mathbf{Tr}(M)^{-1/2}.$$

$\|w\|^*_{\Omega_2}$ is convex as a finite maximum of convex functions, it is clearly homogeneous and as $\mathbf{I} \in \mathcal{M}$ we have $\|w\|^*_{\Omega_2} \implies w = 0$. Hence $\|w\|^*_{\Omega_2}$ is a norm. h^* is then the indicator function of the unit norm ball of $\|w\|^*_{\Omega_2}$.

Fenchel bi-dual of h is then

$$\begin{aligned}
h^{**}(w) &= \sup_{x\in\mathbb{R}^d} w^T x - h^*(x) \\
&= \sup_{x\in\mathbb{R}^d} w^T x - \sum_{M\in\mathfrak{M}} \sup_{\lambda_M\geq 0} \lambda_M(\|Mx\|_2 - \mathbf{Tr}(M)^{1/2}) \\
&= \inf_{(\lambda_M)_{M\in\mathfrak{M}},\, \lambda_M\geq 0} \sum_{M\in\mathfrak{M}} \mathbf{Tr}(M)^{1/2}\lambda_M + \sup_{x\in\mathbb{R}^d} w^T x - \sum_{M\in\mathfrak{M}} \lambda_M \|Mx\|_2 \\
&= \inf_{(\lambda_M)_{M\in\mathfrak{M}},\, \lambda_M\geq 0} \sum_{M\in\mathfrak{M}} \mathbf{Tr}(M)^{1/2}\lambda_M + \sup_{x\in\mathbb{R}^d} w^T x - \sum_{M\in\mathfrak{M}} \lambda_M \sup_{\|a_M\|_2\leq 1} x^T M a_M \\
&= \inf_{\substack{(\lambda_M)_{M\in\mathfrak{M}},\, \lambda_M\geq 0\\ (a_M)_{M\in\mathfrak{M}},\, \|a_M\|_2\leq 1\\ x=\sum_{M\in\mathfrak{M}} \lambda_M M a_M}} \sum_{M\in\mathfrak{M}} \mathbf{Tr}(M)^{1/2}\lambda_M \\
&= \inf_{\substack{(x_M)_{M\in\mathfrak{M}}\\ x=\sum_{M\in\mathfrak{M}} M x_M}} \sum_{M\in\mathfrak{M}} \mathbf{Tr}(M)^{1/2}\|Mx_M\|_2 \\
&= \|w\|_{\Omega_2}.
\end{aligned}$$

Since h^* is the indicator function of the unit ball of $\|w\|^*_{\Omega_2}$, $\|w\|_{\Omega_2}$ is the dual norm of $\|w\|^*_{\Omega_2}$. ■

Computed norm $\|w\|_{\Omega_2}$ appears similar to the grouped norms defined for example by Jacob et al. [2009]. However, to our knowledge, no algorithm can compute the norm or its proximal operator such that its utility in practice is unclear.

Chapter 7

Grouping samples for diverse predictions

Chapter Abstract

In prediction problems involving a large amount of training samples, reducing complexity of the task by clustering data points can improve performance and help interpretation. Here, rather than separating clustering and prediction steps, we study how both can be done simultaneously. We provide a formulation of our problem for regression or classification and present several algorithms to solve it. First, we develop a projected gradient scheme whose core iteration amounts to a k-means step. Then, we present a convex relaxation of the problem using conditional gradient, a.k.a. Frank-Wolfe algorithm, whose core iteration amounts again to a k-means operation. Numerical experiments illustrate the performance of our methods on synthetic data sets.

Introduction

Machine learning aims at analyzing data either to predict observations of future data or to reveal hidden information such as clusters. It has been applied in numerous fields such as computer vision, bio-informatics or economy (see Hastie et al. [2008] for an introduction of the field). While prediction and clustering problems are often analyzed independently, they can benefit from each other.

On one side, if data are clustered beforehand a prediction problem can use this information [Pfeffermann and Nathan, 1981; Graubard and Korn, 1994]. If data could be clustered but clusters are unknown then the prediction problem shall take this information into account. For example, in a regression problem where one half of the data x with attributes y satisfies $y \approx x$ and the other one satisfies $y \approx -x$, a classical algorithm would predict $y \approx 0$ in average, while it may preferable to output two possible answers $y = x$ or $y = -x$ which accounts for the diversity of the data. Note that, in this case, the problem can be seen as a special instance of subspace clustering problem [Vidal, 2011].

On the other side, more and more clustering techniques exist (classical k-means Gan et al. [2007], k-means with kernels [Schölkopf et al., 1998], EM algorithm [Dempster et al., 1977], EM with Bregman divergence [Banerjee et al., 2005], spectral clustering [Von Luxburg, 2007], DIFFRAC [Bach and Harchaoui, 2008], Clusterpath [Hocking et al., 2011] to cite a few) but no common measure of performance allow their comparison. Yet, clustering can be used as a first phase of prediction problem that it simplifies and helps. Performance of the prediction procedure can then be used to compare different clustering of the data.

In this chapter, we investigate how both clustering and prediction can be performed simultaneously. It either can be seen as a prediction problem relaxed to output more than one prediction per sample or as a clustering problem driven by the performance of a prediction task. It applies when data contain hidden information and diverse answers are required as in privacy learning (see Wainwright et al. [2012] and references herein), where personal information influences the predicted attributes but are not revealed.

Diversity learning was studied by Guzman-Rivera et al. [2014] who developed losses to enforce multiple outputs. Here we rather constraint classical learning problems by a clustering one. Mixing clustering and prediction problems is also performed in the mixture of experts framework [Jordan, 1994]. However our setting differs form the latter as we assume that partition of the samples can only be revealed by the observations that have to be predicted and not by the features of the data. Our framework corresponds to the one studied by Zhang [2003] or Bagirov et al. [2013] to perform regression clustering with various models of regression.

We present our framework for regression and classification tasks, it incorporates the partitioning problem of the training samples in the minimization of an empirical loss. First, we propose algorithms to tackle the non-convex resulting problem : an alternate minimization between the partitioning problem and the empirical loss minimization and a projected gradient scheme whose projection step amounts to a clustering problem that can be solved approximatively by k-means++. Then, we present a convex relaxation in the case of a squared loss for which the combinatorial problem can be isolated. We use conditional gradient, a.k.a. Frank-Wolfe algorithm [Frank and Wolfe, 1956; Jaggi, 2013], whose core iteration amounts also to a k-means problem. Finally numerical experiments show the robustness of our method on synthetic data.

7.1 Problem Formulation

We first present our framework for linear regression tasks and then extend its application to classification.

7.1.1 Clustered Regression

We briefly present regression problems, more details are provided in Section 6.1.1. Given n observations $y_1, \dots, y_n \in \mathbb{R}$ from data points $x_1, \dots, x_n \in \mathbb{R}^d$, we are seek-

ing for a vector w that predicts linearly outputs from inputs. A loss function ℓ measures its accuracy error $\ell(w^T x, y)$ on a sample (x, y) such as the squared loss $\ell_{\text{square}}(w^T x, y) = \frac{1}{2}(w^T x - y)^2$. A candidate vector w can be found by minimizing the empirical loss $\frac{1}{n}\sum_{i=1}^n \ell(w^T x_i, y_i)$. In order to prevent over-fitting, a regularizer $R(w)$ of the predictor w is added such as its squared Euclidean norm, i.e. $R_{\text{square}}(w) = \frac{1}{2}\|w\|^2$. The regression problem is then

$$\text{minimize } \frac{1}{n}\sum_{i=1}^n \ell(w^T x_i, y_i) + \lambda R(w)$$

in variable $w \in \mathbb{R}^d$, where $\lambda \geq 0$ is a regularization parameter. Affine regression problems can be treated similarly as detailed in Section 6.1.1.

Here we search for Q linear prediction vectors $v_1, \ldots, v_Q \in \mathbb{R}^d$, each can be used to predict observations of a given sample. Samples can then be partitioned in Q groups $g_1, \ldots, g_Q$ by assigning for each of them the best predictor. When it comes to compute best predictors $v_1, \ldots, v_Q$ on the training data, we face therefore a mix of a partitioning problem and empirical loss minimization. We weight the regularization of each predictor by the number of samples in the group it defines in order to ease the development of our algorithms. Our clustered regression problem reads therefore

$$\text{minimize } \frac{1}{n}\sum_{q=1}^Q \sum_{i \in g_q} \ell\left(v_q^T x_i, y_i\right) + \lambda \sum_{q=1}^Q s_q R(v_q) \tag{7.1}$$

in the partition $G = (g_1, \ldots, g_Q)$ of $\{1, \ldots, n\}$ and prediction vectors $v_1, \ldots, v_Q \in \mathbb{R}^d$, where $s_q = \mathbf{Card}(g_q)$ is the size of group g_q and $\lambda \geq 0$ is a regularization parameter.

By denoting $w_i \in \mathbb{R}^d$ the prediction vector used for sample i such that if $i \in g_q$, $w_i = v_q$, our problem can be rewritten

$$\begin{array}{ll} \text{minimize} & \frac{1}{n}\sum_{i=1}^n \ell\left(w_i^T x_i, y_i\right) + \lambda \sum_{i=1}^n R(w_i) \\ \text{subject to} & w_i = v_q \quad \text{if } i \in g_q, \text{ for all } i \in \{1, \ldots, n\} \end{array} \tag{7.2}$$

in the partition $G = (g_1, \ldots, g_Q)$ of $\{1, \ldots, n\}$ and prediction vectors $w_1, \ldots, w_n \in \mathbb{R}^d$ and $v_1, \ldots, v_Q \in \mathbb{R}^d$, where $\lambda \geq 0$ is a regularization parameter. A partition $G = (g_1, \ldots, g_Q)$ of $\{1, \ldots, n\}$ can then be represented by an assignment matrix $Z \in \{0, 1\}^{n \times Q}$ such that

$$Z_{iq} = \begin{cases} 1 & \text{if} \quad i \in g_q \\ 0 & \text{otherwise.} \end{cases}$$

Partitioning constraints of (7.2) read then $w_i = \sum_{q=1}^Q Z_{iq} v_q$ for all $1 \leq i \leq n$, where Z is the assignment matrix of a partition. Denoting $W = (w_1, \ldots, w_n) \in \mathbb{R}^{d \times n}$ the matrix of individual classifiers, $L(W) = \frac{1}{n}\sum_{i=1}^n \ell\left(w_i^T x_i, y_i\right)$ and $R(W) = \sum_{i=1}^n R(w_i)$, clustered regression problem finally reads

$$\begin{array}{ll} \text{minimize} & L(W) + \lambda R(W) \\ \text{subject to} & W = VZ^T,\ Z \in \{0, 1\}^{n \times Q},\ Z\mathbf{1} = \mathbf{1}, \end{array} \tag{7.3}$$

in variables $W \in \mathbb{R}^{d\times n}$, $V \in \mathbb{R}^{d\times Q}$ and Z, where $\lambda \geq 0$ is a regularization parameter. This algebraic formulation eases the presentation of our algorithmic solutions that we present in next section, before we extend our framework to classification problems.

7.1.2 Clustered classification

We briefly present classification problems, more details are provided in Section 6.1.2. We are given n data points $x_1, \ldots, x_n \in \mathbb{R}^d$ that belong to one of K classes, which is encoded by binary vectors $y_i \in \{-1, 1\}^K$ such that $y_{ik} = 1$ if i^{th} point belongs to class K and -1 otherwise. We search for K linear classifiers $w_1, \ldots, w_K$, forming a matrix of linear classifiers $W \in \mathbb{R}^{d\times K}$ whose classification error on a sample (x, y) is measured by a loss $\ell(W^T x, y)$ such as the squared loss $\ell_{\text{square}}(W^T x, y) = \frac{1}{2}\|W^T x - y\|_2^2$. Candidate classifiers are given by the minimization of the empirical loss $\frac{1}{n}\sum_{i=1}^n \ell(W^T x_i, y_i)$. As for regression, a regularizer $R(W)$ can be added on the linear classifiers such as their squared euclidean norm $R_{\text{square}}(W) = \frac{1}{2}\sum_{k=1}^K \|w_k\|_2^2 = \frac{1}{2}\|W\|_F^2$. Classification problems read then

$$\text{minimize } \frac{1}{n}\sum_{i=1}^n \ell(W^T x_i, y_i) + \lambda R(W)$$

in variable $W \in \mathbb{R}^{d\times K}$, where $\lambda \geq 0$ is a regularization parameter.

As for regression, we search not one but several matrices of classifiers $V_1, \ldots, V_Q \in \mathbb{R}^{d\times K}$ that, for a given sample, outputs diverse estimated classes. Number of possible classifiers Q does naturally not exceed the number of classes K. Samples are then partitioned in Q groups by assigning the best matrix of classifier for each sample. Using same weighted regularization as for regression, our clustered classification problem reads

$$\text{minimize } \frac{1}{n}\sum_{q=1}^Q \sum_{i\in g_q} \ell\left(V_q^T x_i, y_i\right) + \lambda \sum_{q=1}^Q s_q R(V_q) \tag{7.4}$$

in the partition $G = (g_1, \ldots, g_Q)$ of $\{1, \ldots, n\}$ and matrices of classifiers $V_1, \ldots, V_Q \in \mathbb{R}^{d\times K}$, where $s_q = \mathbf{Card}(g_q)$ is the size of group g_q and $\lambda \geq 0$ is a regularization parameter.

By denoting W_i the matrix of classifiers used for sample i such that if $i \in g_q$, $W_i = V_q$ and by using assignment matrices to represent partitions, clustered classification problem reads

$$\begin{array}{ll} \text{minimize} & \frac{1}{n}\sum_{i=1}^n \ell\left(W_i^T x_i, y_i\right) + \lambda \sum_{i=1}^n R(W_i) \\ \text{subject to} & W_i = \sum_{q=1}^Q Z_{iq} V_q, \quad \text{for all } i \in [\![1, n]\!] \\ & Z \in \{0, 1\}^{n\times Q},\ Z\mathbf{1} = \mathbf{1}, \end{array} \tag{7.5}$$

in variables $W_1, \ldots, W_n \in \mathbb{R}^{d\times K}$, $V_1, \ldots, V_Q \in \mathbb{R}^{d\times K}$ and Z, where $\lambda \geq 0$ is a regularization parameter. As for regression this can also applied to compute affine hyperplanes by adding a constant dimension to the data and solve the problem in dimension $d + 1$.

Overall (7.5) possesses same structure as (7.3) by replacing regression vectors with matrices of classifiers. Both problems have convex objectives but non-convex constraints that encode the underlying combinatorial problem. Specifically the feasible set is a union of subspaces generated by assignment matrices of partitions. In the following, we focus on the regression case, classification can be treated with same algorithms by considering matrices instead of vectors.

7.2 Non-convex schemes

We present strategies that tackle directly the non-convex clustered prediction problems.

7.2.1 Greedy algorithm

A first strategy to solve problem (7.1) is to alternate minimization on the predictors of each group and assignment of each point to the group where its loss is smallest. This method is fast but unstable and highly dependent on initialization. However, it can be used to refine the solution of others algorithms proposed below.

7.2.2 Projected gradient descent

Formulation (7.3) enables to develop projected gradient descent on the individual prediction vectors represented by the variable W. Projection of a given $W = (w_1, \dots, w_n) \in \mathbb{R}^{d \times n}$ on the feasible set of (7.3) reads

$$\begin{array}{ll} \text{minimize} & \|W - VZ^T\|_F^2 = \sum_{i=1}^n \|w_i - \sum_{q=1}^Q Z_{iq} v_q\|_2^2 \\ \text{subject to} & Z \in \{0,1\}^{n \times Q}, \quad Z\mathbf{1} = \mathbf{1} \end{array}$$

in variable $V = (v_1, \dots, v_Q) \in \mathbb{R}^{d \times Q}$ and Z. We recognize a k-means problem on the columns of W, i.e.

$$\text{minimize} \sum_{q=1}^{Q} \sum_{i \in g_q} \|w_i - v_q\|_2^2, \tag{7.6}$$

in variables $v_1, \dots, v_Q \in \mathbb{R}^d$ that are the centroids of the clusters and $G = \{g_1, \dots, g_Q\}$ a partition of $\{1, \dots, n\}$. Careful initialization as made in k-means++ [Arthur and Vassilvitskii, 2007] offers logarithmic approximations to the problem such that approximate projection is available. Given a matrix W, whose columns we want to cluster in Q groups, we denote by $[Z, V] = \text{k-means}(W, Q)$ respectively the assignment matrix and the matrix of centroids output by a clustering algorithm. A projected gradient scheme for problem (7.3) is described in Algorithm 8.

Algorithm 8 Projected gradient descent for clustered regression

Inputs: Data (X, Y), desired number of groups Q, step size γ_t
Initialize $W_0 \in \mathbb{R}^{d\times K}$
for t = 1,. . . ,T **do**
$W_{t+1/2} = W_t - \gamma_t(\nabla L(W_t) + \nabla R(W_t))$
$[Z_{t+1}, V_{t+1}] = \text{k-means}(W_{t+1/2}, Q)$
$W_{t+1} = V_{t+1} Z_{t+1}^T$
end for
Output: $\hat{W} = W_T$

In practice, we stop the algorithm when change in objective values are below some prescribed threshold ε. We use a backtracking line search on the stepsize γ_t that guarantees decreasing of the objective. Precisely, at each iteration if

$$\bar{W}_{t+1} \quad = \text{k-means}\left(W_t - \gamma_t(\nabla L(W_t) + \nabla R(W_t)), Q\right)$$

decreases the objective value we keep them and we increase the stepsize by a constant factor $\gamma_{t+1} = \alpha\gamma_t$ with $\alpha > 1$. If $\bar{W}_{t+1}$ increases the objective value we decrease the stepsize by a constant factor $\gamma_t \leftarrow \beta\gamma_t$, with $\beta < 1$, compute new $\bar{W}_{t+1}$ and iterate this operation until $\bar{W}_{t+1}$ decreases the objective value or the stepsize reaches the stopping value ε used as a stopping criterion on the objective values. We observed better results with this line search than with constant stepsize.

Using this strategy we observed convergence of the projected gradient algorithm in less than 100 iterations even for large n which makes it scalable. The complexity of its core operations amounts indeed to k-means operations. k-means++ initialization costs $O(Q^2nd)$ operations and standard alternating minimization approximates the k-means operation at a cost of $O(TQnd)$, where T, the number of alternating steps, is generally small.

7.3 Convex relaxation

Difficulty of clustered regression problems (7.1) lies in its underlying combinatorial nature that can be isolated in the case of a squared loss and squared regularizer. In this case problem (7.1) reads

$$\text{minimize } \frac{1}{2n}\sum_{q=1}^{Q}\sum_{i\in g_q}\left(v_q^T x_i - y_i\right)^2 + \frac{\lambda}{2}\sum_{q=1}^{Q} s_q\|v_q\|_2^2 \tag{7.7}$$

in the partition $G = (g_1, \ldots, g_Q)$ of $\{1, \ldots, n\}$ and prediction vectors $v_1, \ldots, v_Q \in \mathbb{R}^d$, where $s_q = \mathbf{Card}(g_q)$ is the size of group g_q and $\lambda \geq 0$ is a regularization parameter. In the following, we denote $X = (x_1, \ldots, x_n)^T \in \mathbb{R}^{n\times d}$ the matrix of data points and $y = (y_1, \ldots, y_n)^T \in \mathbb{R}^n$ the vector of observations.

7.3.1 Simplified formulation for squared loss

The squared loss has the advantage to provide analytic solutions for prediction problems in the continuous variables. Resulting problem on partitions can be expressed in terms of normalized equivalence matrices presented in Section 5.2. Following proposition details the simplified formulations of the clustered prediction problem for squared loss and squared regularizer.

Proposition 7.3.1. *Problem (7.7) is equivalent to*

$$\begin{array}{ll} \text{minimize} & \frac{1}{2n} y^T (\mathbf{I} + \frac{1}{\lambda n} X X^T \circ M)^{-1} y \\ \text{subject to} & M \in \mathcal{M} \end{array} \tag{7.8}$$

where $\mathcal{M} = \{M : M = Z(Z^T Z)^\dagger Z^T, Z \in \{0,1\}^{n \times Q}, Z\mathbf{1} = \mathbf{1}\}$ *is the set of equivalence matrices.*

Proof. We fix a partition $G = (g_1, \ldots, g_Q)$ and define for each group $g_q = \{k_1, \ldots, k_{s_q}\} \subset \{1, \ldots, n\}$, the matrix $E \in \{0,1\}^{s_q \times n}$ that picks the s_q points of g_q, *i.e.* $(E_q)_{ij} = 1$ if $j = k_i$ and 0 otherwise. Therefore $y_q = E_q y \in \mathbb{R}^{s_q}$ and $X_q = E_q X \in \mathbb{R}^{s_q \times d}$ are respectively the vector of labels and the matrix of sample vectors of the group g_q. We naturally have $E_q E_q^T = \mathbf{I}$ as rows of E_q are orthonormal and $E_q^T E_q$ is a diagonal matrix where $Z_q = \mathbf{diag}(E_q^T E_q) \in \{0,1\}^n$ is the assignment vector in group g_q, *i.e.* $(Z_q)_j = 1$ if $j \in g_q$ and 0 otherwise. $Z = (Z_1, \ldots, Z_Q)$ is therefore an assignment matrix for the partition G.

Minimizing in v and using the Sherman-Woodbury-Morrison formula, we obtain a function of the partition

$$\begin{aligned} \psi(G) &= \min_{v_1,\ldots,v_Q} \frac{1}{2n} \sum_{q=1}^{Q} \|y_q - X_q v_q\|_2^2 + \frac{\lambda}{2} \sum_{q=1}^{Q} s_q \|v_q\|_2^2 \\ &= \frac{1}{2n} \sum_{q=1}^{Q} \left(\|y_q\|_2^2 - y_q^T X_q (s_q \lambda n \mathbf{I} + X_q^T X_q)^{-1} X_q^T y_q \right) \\ &= \frac{1}{2n} \sum_{q=1}^{Q} y_q^T (\mathbf{I} + \frac{1}{s_q \lambda n} X_q X_q^T)^{-1} y_q. \end{aligned}$$

Formulating terms of the sum as solutions of an optimization problem, we get

$$\begin{aligned} \psi(G) &= \frac{1}{2n} \sum_{q=1}^{Q} \max_{\alpha_q \in \mathbb{R}^{s_q}} -\alpha_q^T (\mathbf{I} + \frac{1}{s_q \lambda n} X_q X_q^T) \alpha_q + 2 y_q^T \alpha_q \\ &= \frac{1}{2n} \max_{\substack{\alpha = (\alpha_1; \ldots; \alpha_Q) \\ \alpha_q \in \mathbb{R}^{s_q}}} \sum_{q=1}^{Q} -\alpha_q^T (\mathbf{I} + \frac{1}{s_q \lambda n} X_q X_q^T) \alpha_q + 2 y_q^T \alpha_q, \end{aligned}$$

where $(\alpha_1; \ldots; \alpha_Q) = (\alpha_1^T, \ldots, \alpha_Q^T)^T$ stacks vectors α_q in one vector of size $\sum_{q=1}^{Q} s_q =$

n. Using that $E = (E_1; \ldots; E_Q) = (E_1^T, \ldots, E_Q^T)^T \in \{0,1\}^{n \times n}$ is an orthonormal matrix, we make the change of variable $\beta = E^T\alpha$ (and so $\alpha = E\beta$) such that for $\alpha = (\alpha_1; \ldots; \alpha_Q)$, $\alpha_q \in \mathbb{R}^{s_q}$, $\alpha_q = E_q\beta$. Decomposing X_q and y_q and using $E_q^T E_q = \mathbf{diag}(Z_q)$, we get

$$\begin{aligned}
\psi(G) &= \frac{1}{2n} \max_{\beta \in \mathbb{R}^n} \sum_{q=1}^{Q} -\beta^T E_q^T (\mathbf{I} + \frac{1}{s_q \lambda n} X_q X_q^T) E_q \beta + 2 y_q^T E_q \beta \\
&= \frac{1}{2n} \max_{\beta \in \mathbb{R}^n} \sum_{q=1}^{Q} -\beta^T E_q^T (\mathbf{I} + \frac{1}{s_q \lambda n} E_q X X^T E_q^T) E_q \beta + 2 y^T E_q^T E_q \beta \\
&= \frac{1}{2n} \max_{\beta \in \mathbb{R}^n} \sum_{q=1}^{Q} -\beta^T \mathbf{diag}(Z_q)\beta - \frac{1}{s_q \lambda n} \beta^T \mathbf{diag}(Z_q) X X^T \mathbf{diag}(Z_q)\beta + 2 y^T \mathbf{diag}(Z_q)\beta.
\end{aligned}$$

For q fixed, $\left(\frac{1}{s_q}\mathbf{diag}(Z_q) X X^T \mathbf{diag}(Z_q)\right)_{ij} = \frac{1}{s_q} x_i^T x_j$ if $(i,j) \in g_q$ and 0 otherwise. So

$$\sum_{q=1}^{Q} \frac{1}{s_q} \mathbf{diag}(Z_q) X X^T \mathbf{diag}(Z_q) = X X^T \circ M,$$

where $M = Z(Z^T Z)^\dagger Z^T$ is the normalized equivalence matrix of the partition G and $\circ$ denotes the Hadamard product. Using $\sum_{q=1}^{Q} \mathbf{diag}(Z_q) = \mathbf{I}$, we finally get a function of the equivalence matrix

$$\begin{aligned}
f(M) &= \frac{1}{2n} \max_{\beta \in \mathbb{R}^n} -\beta^T (\mathbf{I} + \frac{1}{\lambda n} X X^T \circ M)\beta + 2 y^T \beta \\
&= \frac{1}{2n} y^T (\mathbf{I} + \frac{1}{\lambda n} X X^T \circ M)^{-1} y.
\end{aligned}$$

■

The resulting problem is still non-convex due to the combinatorial nature of the set of equivalence matrices. However one can then relax the problem by optimizing on its convex hull as presented in Section 5.3.2.

7.3.2 Conditional gradient algorithm application

We detail the linear minimization oracle used by Frank-Wolfe Algorithm 5 in this setting. Denote the objective function of problem (7.8) by

$$f(M) \triangleq \frac{1}{2n} y^T (\mathbf{I} + \frac{1}{\lambda n} X X^T \circ M)^{-1} y. \tag{7.9}$$

Its gradient at a given $M \in \mathcal{M}$ is

$$\nabla f(M) = -\frac{1}{2\lambda n^2} X X^T \circ \left((\mathbf{I} + \frac{1}{\lambda n} X X^T \circ M)^{-1} y y^T (\mathbf{I} + \frac{1}{\lambda n} X X^T \circ M)^{-1} \right). \tag{7.10}$$

Observe that $-\nabla f(M)$ is a semi-definite positive matrix with squared root

$$U = \frac{1}{\sqrt{2\lambda n}} \mathbf{diag}\left((\mathbf{I} + \frac{1}{\lambda n} XX^T \circ M)^{-1} y\right) X \in \mathbb{R}^{n \times d}.$$

The linear minimization oracle to minimize $f(M)$ over the convex hull of the set of normalized equivalence matrices can then be computed as follows

$$\begin{aligned}
\underset{S \in \text{hull}(\mathcal{M})}{\text{argmin}} \langle S, \nabla f(M) \rangle &\overset{\vartheta_1}{=} \underset{S \in \mathcal{M}}{\text{argmin}} \mathbf{Tr}(S^T \nabla f(M)) \\
&= \underset{S \in \mathcal{M}}{\text{argmin}} - \mathbf{Tr}(SUU^T) \\
&= \underset{S \in \mathcal{M}}{\text{argmin}} \mathbf{Tr}((\mathbf{I} - S)UU^T)) \\
&\overset{\vartheta_2}{=} \underset{S \in \mathcal{M}}{\text{argmin}} \|U - SU\|_F^2. \qquad (7.11)
\end{aligned}$$

In ϑ_1, we used that $\mathcal{M}$ is a set of atoms, so its convex hull is a polytope and linear minimization on it is equivalent to linear minimization on its vertices, i.e., $\mathcal{M}$. In ϑ_2, we used that normalized equivalence matrices are orthogonal projectors, so $I - S$ is also an orthogonal projector. Now we observe that (7.11) is a k-means problem as presented in Section 5.2.2. Therefore solving a k-means problem on the rows of the squared root matrix U of $-\nabla f(M)$ offers a solution to the linear minimization oracle. Careful initialization as made in k-means++ [Arthur and Vassilvitskii, 2007] offers logarithmic approximations to the problem.

A conditional gradient method with approximate oracle can then be applied to solve a relaxed version of (7.8). Once done, it remains to provide an approximate feasible solution for the original constraints. Two projections of the relaxed version are possible: either finding the closest normalized equivalence matrix in Frobenius norm or computing the point that minimizes the gradient of the relaxed solution, i.e. computing its linear minimization oracle. In practice we chose second solution as it provided better results. Notice that the k-means operation used for the linear minimization provides not only a normalized equivalence matrix but also a corresponding partition that leads to the optimal predictors $v_1, \ldots, v_Q$ as

$$\begin{aligned}
v_q &= (n\lambda s_q \mathbf{I} + X_q^T X_q)^{-1} X_q^T y_q \\
&= (n\lambda s_q \mathbf{I} + X^T E_q^T E_q X)^{-1} X^T E_q^T E_q y \qquad (7.12) \\
&= (n\lambda s_q \mathbf{I} + X^T \mathbf{diag}(Z_q) X)^{-1} X^T \mathbf{diag}(Z_q) y,
\end{aligned}$$

where X_q, y_q, Z_q and E_q are defined in the proof of Proposition 7.3.1.

To summarize, our convex relaxation for regression with grouped features is presented in Algorithm 9. We denote by $Z = \text{k-means}(U, Q)$ an assignment matrix solution of the k-means problem that cluster rows of U in Q groups.

Algorithm 9 Convex relaxation for regression with grouped features

Inputs: Data (X, Y), desired number of groups Q, target precision ε
Initialize $M_0 \in \mathcal{M}$
for $t = 0, \ldots$ **do**
Compute $-\nabla f(M_t)$ in (7.10) and its squared root U
Get linear minimization oracle by computing

$$\begin{aligned} Z_t &= \text{k-means}(U, Q) \\ S_t &= Z_t (Z_t^T Z_t)^\dagger Z_t^T \end{aligned}$$

if Tr $\left((M_t - S_t)^T \nabla f(M_t)\right) \leq \varepsilon$ **then** Stop **end if**
Set $M_{t+1} = M_t + \frac{1}{t+2}(S_t - M_t)$
end for
Output: Optimal predictors $v_1, \ldots, v_Q$ in (7.12) from partition given by Z_t.

We briefly examine the complexity of Algorithm 6. To get linear minimization oracle we use a k-means++ initialization that costs $O(Q^2nd)$ operations and standard alternating minimization approximates then the k-means operation at a cost of $O(TQnd)$, where T is the number of alternating steps, that is generally small. The squared root computation is directly given by the computation of $\nabla f(M)$. However this gradient itself requires the inversion of a matrix of size $n \times n$. This burdens its implementation for big data sets. However a few iterations of this algorithm can give a good initialization for non-convex approaches of the problem.

7.4 Numerical experiments

We test the robustness of our method when data can be clustered using a few features. We generate n data points (x_i, y_i) for $i = 1, \ldots, n$, with $x_i \in \mathbb{R}^d$, $d = 10$, and $y_i \in \mathbb{R}$, divided in $Q = 3$ clusters corresponding to regression tasks with weight vectors v_q. Regression labels for points x_i in group g_q are given by $y_i = v_q^T x_i + \eta_y$, where $\eta \sim \mathcal{N}(0, \sigma^2)$. We test the robustness of the algorithms to the addition of noisy dimensions by completing x_i with δ dimensions of noise $\theta \sim \mathcal{N}(0, \tau^2)$. For testing the models we take the difference between the true label and the best prediction such that the mean square error is given by

$$\frac{1}{2n} \sum_{i=1}^{n} \min_{q=1\ldots,Q} (y_i - v_q^T x_i)^2. \tag{7.13}$$

The results are reported in Table 7.1 where the intrinsic dimension is 10 and the proportion of dimensions of noise $\delta/(d+\delta)$ increases. On the algorithmic side, "Oracle" refers to the least-squares fit given the true assignments, which can be seen as the best achievable error rate, AM refers to alternate minimization, PG refers to projected gradient with squared loss, CG refers to conditional gradient and RC to regression clustering as proposed by Zhang [2003], implemented using the Harmonic K-means

formulation. PG, CG and RC were followed by AM refinement. 1000 points were used for training, 100 for testing. The regularization parameters were 5-fold cross-validated using a logarithmic grid. Noise on labels is $\sigma = 10^{-1}$ and noise on added dimensions is $\tau = 1$. Results were averaged over 50 experiments with figures after the $\pm$ sign corresponding to one standard deviation.

	p = 0	p = 0.25	p = 0.5	p = 0.75	p = 0.9
Oracle	0.52±0.08	0.55±0.07	0.55±0.10	0.58±0.09	0.71±0.11
AM	**0.52**±0.08	**0.55**±0.07	5.57±4.11	6.93±14.39	101.08±55.49
PG	1.53±7.13	3.98±17.65	**3.20**±13.23	5.64±20.50	91.33±39.32
CG	0.87±2.45	1.16±4.29	3.64±11.02	**5.43**±14.33	91.19±53.00
RC	**0.52**±0.08	**0.55**±0.07	5.59±20.27	13.45±28.76	**59.19**±37.97

Table 7.1 – Test mean square error given by (7.13) along proportion of added dimensions of noise $p = \delta/(d + \delta)$.

All algorithms perform similarly, RC and AM get better results without added noise. None of the present algorithms get a significantly better behavior with a majority of noisy dimensions.

7.5 Conclusion

We developed a framework for diversity learning that finds groups of samples which share same predictor. Projected gradient scheme offers a scalable algorithmic approach for any loss, while convex relaxation for squared loss gives a first good estimate. However core inner steps of these approaches are only approximated by k-means++, a finer analysis would require to both tackle non-convexity and inexact oracles. Yet, this framework may find its application in privacy learning to leverage hidden information.

Chapter 8

Clustered multi-task

Chapter Abstract

Multi-task learning aim at solving several prediction tasks simultaneously with the hope that they can share information to improve overall performance. Based on previous clustered multi-task learning framework of Jacob et al. [2009], we investigate how similar tasks can be clustered to exploit common information. To this end, a clustering penalty on the prediction vectors is added to the empirical loss minimization problem. Here, we isolate the clustering problem and treat it with various optimization strategies. First, we present a projected gradient descent whose core iteration amounts to a k-means problem on the predictors. Then, we show that in case of a squared loss, clustered multi-task reduces to a k-means problem. Numerical experiments detail the performance of our approach on text classification.

Introduction

A supervised learning problem aims at fitting a model on a set of training data that can generalize to future samples. Generalization performance naturally increases with the number of training samples but can also benefit from the number of tasks performed simultaneously on the data [Maurer et al., 2016]. Similar tasks can indeed share information to improve overall performance [Argyriou et al., 2008; Evgeniou and Pontil, 2004; Pan and Yang, 2010]. For example, in image classification, classifiers associated with different species of cats should be quite similar, but well separated from classifiers associated with cars. This observation motivated Jacob et al. [2009] to regularize multi-task learning problems by a clustering penalty that groups similar tasks. It led to further developments in the transfer learning literature [Zhang and Yeung, 2010; Kumar and Daume, 2012; Ciliberto et al., 2015; Zhou et al., 2011].

In this work, we investigate new optimization procedures of the clustered multi-task of Jacob et al. [2009]. We use same clustering penalty that can be decomposed in three terms : overall regularization of the tasks, regularization of the variance between clusters and regularization of the variance within clusters. However, here,

we isolate the clustering problem and do not relax it. First, we develop a projected gradient descent, that clusters predictors at each prediction step. Then, for squared loss we isolate the underlying partitioning problem on the tasks and show that it amounts to a k-means problem. We illustrate the performance on our approach on topics classification of a corpus of texts.

8.1 Problem Formulation

For simplicity, we illustrate the case of multi-category classification, which can be extended to the general multi-task setting. We briefly present classification problems, more details are provided in Section 6.1.2. We are given n data points $x_1, \ldots, x_n \in \mathbb{R}^d$ that belong to one of K classes, which is encoded by binary vectors $y_i \in \{-1, 1\}^K$ such that $y_{ik} = 1$ if i^{th} point belongs to class K and -1 otherwise. We search for K linear classifiers $w_1, \ldots, w_K$, forming a matrix of linear classifiers $W \in \mathbb{R}^{d \times K}$ whose classification error on a sample (x, y) is measured by a loss $\ell(W^T x, y)$ such as the squared loss $\ell_{\text{square}}(W^T x, y) = \frac{1}{2}\|W^T x - y\|_2^2$. Candidate classifiers are given by the minimization of the empirical loss

$$L(W) = \frac{1}{n} \sum_{i=1}^{n} \ell(W^T x_i, y_i).$$

Various strategies are used to leverage the information coming from related tasks, such as low rank [Argyriou et al., 2008] or structured norm penalties [Ciliberto et al., 2015] on the matrix of classifiers W. Here we follow the clustered multitask setting introduced in Jacob et al. [2009]. Namely we add a penalty ω on the classifiers $(w_1, \ldots, w_K)$ which enforce them to be clustered in Q groups $g_1, \ldots, g_Q$, centered around Q points $v_1, \ldots, v_Q \in \mathbb{R}^d$. By denoting $G = (g_1, \ldots g_Q)$ the partition of the K tasks, $s_q = \textbf{Card}(g_q)$ the size of the group g_q and $V = (v_1 \ldots, v_Q) \in \mathbb{R}^{d \times Q}$ the matrix of centroids, this penalty can be decomposed in

— A measure of the **variance within clusters**,

$$\omega_{within}(W, V, G) = \frac{1}{2} \sum_{q=1}^{Q} \sum_{i \in g_q} ||w_i - v_q||_2^2$$

— A measure of the **variance between clusters**,

$$\omega_{between}(V, G) = \frac{1}{2} \sum_{q=1}^{Q} s_q ||v_q - \bar{v}||_2^2$$

— A measure of the **norm of the barycenter of centers** $\bar{v} = \frac{1}{K} \sum_{q=1}^{Q} s_q v_q$,

$$\omega_{mean}(V, G) = \frac{K}{2} ||\bar{v}||_2^2$$

The total penalty reads

$$\omega(W, V, G) = \lambda_w \omega_{within}(W, V, G) + \lambda_b \omega_{between}(V, G) + \lambda_m \omega_{mean}(V, G),$$

where $\lambda_w, \lambda_b, \lambda_m$ are non-negative regularization parameters. It is illustrated in Figure 8-1.

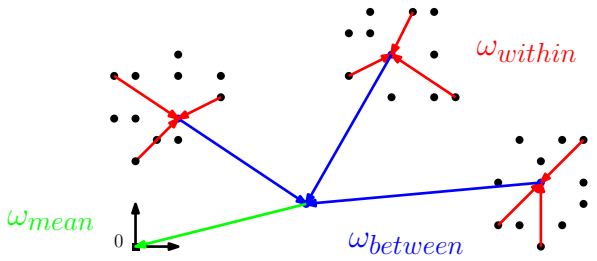

Figure 8-1 – Decomposed clustering penalty.

The partition $G = (g_1, \ldots, g_Q)$ of the K tasks can then be represented by an assignment matrix $Z \in \{0, 1\}^{K \times Q}$, whose columns index tasks and rows groups such that

$$Z_{kq} = \begin{cases} 1 & \text{if} \quad k \in g_q \\ 0 & \text{otherwise.} \end{cases}$$

To represent the barycenter of the predictors, we use the orthogonal projection $\Pi_K = 1_K 1_K^T / K \in \mathbb{R}^{K \times K}$ on the vector $\mathbf{1}_K$ of all ones in dimension K . Given Q centroids forming a matrix $V = (v_1, \ldots, v_Q) \in \mathbb{R}^{d \times Q}$ and a partition $G = (g_1, \ldots, g_Q)$ represented by an assignment matrix $Z \in \{0, 1\}^{K \times Q}$ such that $Z\mathbf{1} = \mathbf{1}$, previous penalties read

$$\Omega_{within}(W, V, Z) = \frac{1}{2} \sum_{q=1}^{Q} \sum_{i \in g_q} ||w_i - v_q||_2^2 = \frac{\lambda_w}{2} ||W - VZ^T||_F^2.$$

$$\Omega_{between}(V, Z) = \frac{1}{2} \sum_{q=1}^{Q} s_q ||v_q - \bar{v}||_2^2 = \frac{\lambda_b}{2} \mathbf{Tr}(VZ^T(\mathbf{I}_K - \Pi_K)ZV^T),$$

$$\Omega_{mean}(V, Z) = K \frac{1}{2} ||\bar{v}||_2^2 = \frac{\lambda_m}{2} \mathbf{Tr}(VZ^T \Pi Z V^T),$$

Using $U = VZ^T$, the total penalty can then be written

$$\Omega(W, U) = \frac{\lambda_w}{2} ||W - U||_F^2 + \frac{\lambda_b}{2} \mathbf{Tr}(U(\mathbf{I}_K - \Pi_K)U^T) + \frac{\lambda_m}{2} \mathbf{Tr}(U \Pi_K U^T). \tag{8.1}$$

where $\lambda_w, \lambda_b, \lambda_m$ are non-negative regularization parameters. Overall, clustered multitask learning problem reads then

$$\begin{array}{ll} \text{minimize} & L(W) + \Omega(W, U) \\ \text{s.t.} & U = VZ^T, \quad Z \in \{0, 1\}^{K \times Q}, \quad Z\mathbf{1} = \mathbf{1}, \end{array} \tag{8.2}$$

in variables $W \in \mathbb{R}^{d\times K}, U \in \mathbb{R}^{d\times K}, V \in \mathbb{R}^{d\times Q}$ and Z. This formulation exhibits a convex objective under non-convex constraints that encode the clustering problem. Several optimization procedures can then be used to treat these constraints.

8.2 Projected gradient descent

Formulation (8.2) enables to develop projected gradient descent on the auxiliary variable U. Projection of a given $U = (u_1, \dots, u_K) \in \mathbb{R}^{d\times K}$ on the feasible set of (8.2) reads

$$\begin{array}{ll} \text{minimize} & \|U - VZ^T\|_F^2 = \sum_{i=1}^K \|u_i - \sum_{q=1}^Q Z_{iq} v_q\|_2^2 \\ \text{subject to} & Z \in \{0,1\}^{K\times Q}, \quad Z\mathbf{1} = \mathbf{1} \end{array}$$

in variable $V = (v_1, \dots, v_Q) \in \mathbb{R}^{d\times Q}$ and Z. We recognize a k-means problem on the columns of U, i.e.

$$\text{minimize} \sum_{q=1}^{Q} \sum_{i\in g_q} \|u_i - v_q\|_2^2, \tag{8.3}$$

in variables $v_1, \dots, v_Q \in \mathbb{R}^d$ that are the centroids of the clusters and $G = (g_1, \dots, g_Q)$ a partition of $\{1, \dots, K\}$. Careful initialization as made in k-means++ [Arthur and Vassilvitskii, 2007] offers logarithmic approximations to the problem such that approximate projection is available. Given a matrix W, whose columns we want to cluster in Q groups, we denote by $[Z, V] = \text{k-means}(W, Q)$ respectively the assignment matrix and the matrix of centroids output by a clustering algorithm. A projected gradient scheme for problem (8.2) is described in Algorithm 10.

Algorithm 10 Projected gradient descent for clustered multi-task

Inputs: Data (X, Y), desired number of groups Q, step size γ_t
Initialize $W_0 \in \mathbb{R}^{d\times K}$
for t = 1,...,T **do**
  $W_{t+1} = W_t - \gamma_t \left(\nabla L(W_t) + \nabla\Omega(W_t, U_t)\right)$
  $U_{t+1/2} = U_t - \gamma_t(\nabla\Omega(W_t, U_t))$
  $[Z_{t+1}, V_{t+1}] = \text{k-means}(U_{t+1/2}, Q)$
  $U_{t+1} = V_{t+1} Z_{t+1}^T$
end for
Output: $\hat{W} = W_T$

In practice we stop the algorithm when change in objective values are below some prescribed threshold ε. We use a backtracking line search on the stepsize γ_t that guarantees decreasing of the objective. Precisely, at each iteration if

$$\begin{aligned} \bar{W}_{t+1} &= W_t - \gamma_t(\nabla L(W_t) + \nabla\Omega(W_t, U_t)) \\ \bar{U}_{t+1} &= \text{k-means}\left(U_t - \gamma_t \nabla\Omega(W_t, U_t), Q\right) \end{aligned}$$

decrease the objective value we keep them and we increase the stepsize by a constant factor $\gamma_{t+1} = \alpha\gamma_t$ with $\alpha > 1$. If $\bar{W}_{t+1}, \bar{U}_{t+1}$ increase the objective value we decrease

the stepsize by a constant factor $\gamma_t \leftarrow \beta\gamma_t$, with $\beta < 1$, compute new $\bar{W}_{t+1}, \bar{U}_{t+1}$ and iterate this operation until $\bar{W}_{t+1}, \bar{U}_{t+1}$ decrease the objective value or the stepsize reaches the stopping value ε used as a stopping criterion on the objective values. We observed better results with this line search than with constant stepsize.

Using this strategy we observed convergence of the projected gradient algorithm in less than 100 iterations even for large dimension (big n, d or K), which makes it scalable. The complexity of its core operations amounts indeed to k-means operations. k-means++ initialization costs $O(Q^2 n)$ operations and standard alternating minimization approximates the k-means operation at a cost of $O(TQn)$, where T, the number of alternating steps, is generally small.

8.3 Clustered multitask with squared loss

Difficulty of clustered multitask problems (8.2) lies in its underlying combinatorial nature that can be isolated in the case of a squared loss. We first detail the simplified formulation and then show how it reduces to a k-means problem.

8.3.1 Simplification for squared loss

When simplifying the clustered multitask problem, partitions are expressed in terms of normalized equivalence matrices presented in Section 5.2. First, simplification can be made on the clustering penalty (8.1) by minimizing on the variable V for W and Z fixed as detailed in the following lemma.

Lemma 8.3.1. *For given matrix of classifier $W \in \mathbb{R}^{d\times K}$ and assignment matrix $Z \in \{0,1\}^{K\times Q}$, clustering penalty (8.1) with regularization parameters λ_m, λ_b and λ_w simplifies by minimizing in the matrix of centroids $V \in \mathbb{R}^{d\times Q}$ as*

$$\min_V \Omega(W, VZ^T) = \frac{\lambda_w}{2} \mathbf{Tr}\left(W\left(\mathbf{I}_K - \rho_{wb}(M - \Pi_K) - \rho_{wm}\Pi_K\right)W^T\right)$$

where $\rho_{wb} = \frac{\lambda_w}{\lambda_w+\lambda_b}$, $\rho_{wm} = \frac{\lambda_w}{\lambda_w+\lambda_m}$ and $M = Z(Z^TZ)^\dagger Z^T$ is the normalized equivalence matrix of the partition represented by Z.

Proof.

$$\begin{aligned}
\min_V \Omega(W, VZ^T) &= \min_V \frac{\lambda_w}{2}\|W - VZ^T\|_F^2 + \frac{\lambda_b}{2}\mathbf{Tr}(VZ^T(\mathbf{I}_K - \Pi_K)ZV^T) + \frac{\lambda_m}{2}\mathbf{Tr}(VZ^T\Pi_K ZV^T) \\
&= \min_V \frac{1}{2}\mathbf{Tr}\left(V\left((\lambda_w+\lambda_b)Z^TZ + (\lambda_m - \lambda_b)Z^T\Pi_K Z\right)V^T\right) - \lambda_w\mathbf{Tr}(V^TWZ) \\
&\quad + \frac{\lambda_w}{2}\|W\|_F^2 \\
&= -\frac{\lambda_w^2}{2}\mathbf{Tr}\left(WZ\left((\lambda_w+\lambda_b)Z^TZ + (\lambda_m-\lambda_b)Z^T\Pi_K Z\right)^{-1}Z^TW^T\right) + \frac{\lambda_w}{2}\|W\|_F^2.
\end{aligned}$$

Denote $s = (s_1, \ldots, s_Q) \in \mathbb{R}^Q$ the vector of the size of the groups $g_1, \ldots, g_Q$ that can be computed from the assignment matrix Z as $s = Z^T\mathbf{1}_K$, $s^{\frac{1}{2}}$ the vector whose

coordinates are $\sqrt{s}_q$ and $s^{-\frac{1}{2}}$ the vector whose coordinates are $\frac{1}{\sqrt{s}_q}$ if $s_q \neq 0$ and 0 otherwise. Inversion in the precedent formula reads

$$\begin{aligned}
J^{-1} &= \left((\lambda_w + \lambda_b) Z^T Z + (\lambda_m - \lambda_b) Z^T \Pi_K Z\right)^{-1} \\
&= \left((\lambda_w + \lambda_b)\, \mathbf{diag}(s) + (\lambda_m - \lambda_b) \frac{s s^T}{K}\right)^{-1} \\
&\overset{\vartheta_1}{=} \frac{1}{\lambda_w + \lambda_b}\, \mathbf{diag}(s^{-\frac{1}{2}}) \left(\mathbf{I}_Q + \frac{\lambda_m - \lambda_b}{\lambda_w + \lambda_b} \frac{s^{\frac{1}{2}} s^{\frac{1}{2}^T}}{K}\right)^{-1} \mathbf{diag}(s^{-\frac{1}{2}}) \\
&\overset{\vartheta_2}{=} \frac{1}{\lambda_w + \lambda_b}\, \mathbf{diag}(s^{-\frac{1}{2}}) \left(\mathbf{I}_Q - \frac{\lambda_m - \lambda_b}{\lambda_w + \lambda_m} \frac{s^{\frac{1}{2}} s^{\frac{1}{2}^T}}{m}\right) \mathbf{diag}(s^{-\frac{1}{2}}) \\
&= \frac{1}{\lambda_w + \lambda_b} (Z^T Z)^\dagger - \frac{\lambda_m - \lambda_b}{(\lambda_w + \lambda_m)(\lambda_w + \lambda_b)} \frac{\mathbf{1}_Q \mathbf{1}_Q^T}{K} \\
&= \frac{1}{\lambda_w + \lambda_b} \left((Z^T Z)^\dagger - \frac{\mathbf{1}_Q \mathbf{1}_Q^T}{K}\right) + \frac{1}{\lambda_w + \lambda_m} \frac{\mathbf{1}_Q \mathbf{1}_Q^T}{K},
\end{aligned}$$

where we used in ϑ_1 that $Z^T Z = \mathbf{diag}(s)$ and in ϑ_2 that $P = \frac{s^{\frac{1}{2}} s^{\frac{1}{2}^T}}{K} = \frac{s^{\frac{1}{2}} s^{\frac{1}{2}^T}}{\|s^{\frac{1}{2}}\|_2^2}$ is a projector and therefore $(I + \alpha P)^{-1} = I - \frac{\alpha}{\alpha+1} P$.

Now introducing the equivalence matrix $M = Z(Z^T Z)^\dagger Z^T$, and using that $Z \frac{\mathbf{1}_Q \mathbf{1}_Q^T}{m} Z^T = \Pi_K$, we finally obtain

$$\begin{aligned}
\min_V \Omega(W, V Z^T) &= \frac{\lambda_w}{2} \mathbf{Tr}(W W^T) - \frac{\lambda_w^2}{2} \mathbf{Tr}\left(W \left(\frac{1}{\lambda_w + \lambda_b}(M - \Pi_K) + \frac{1}{\lambda_w + \lambda_m} \Pi_K\right) W^T\right) \\
&= \frac{\lambda_w}{2} \mathbf{Tr}\left(W \left(\mathbf{I}_K - \rho_{wb}(M - \Pi_K) - \rho_{wm} \Pi_K\right) W^T\right),
\end{aligned}$$

where $\rho_{wb} = \frac{\lambda_w}{\lambda_w + \lambda_b}$ and $\rho_{wm} = \frac{\lambda_w}{\lambda_w + \lambda_m}$. ■

The squared loss has now the advantage to provide analytic solutions for multitask problems in the variable W which simplifies clustered multitask problem (8.2) as shown in following proposition. Training data is there represented by the design matrix $X = (x_1, \ldots, x_n)^T \in \mathbb{R}^{n \times d}$ and the matrix of labels $(y_1, \ldots, y_n)^T \in \mathbb{R}^{n \times K}$.

Proposition 8.3.2. *Clustered multitask problem (8.2) with regularization parameters λ_m, λ_b and λ_w is equivalent to*

$$\begin{array}{ll} \text{minimize} & \mathbf{Tr}(A^T M) \\ \text{subject to} & M \in \mathcal{M} \end{array} \tag{8.4}$$

where

$$A = Y^T X \left((X^T X + n\lambda_w \mathbf{I}_d)^{-1} - (X^T X + n\lambda_w (1-\rho_{wb})\mathbf{I}_d)^{-1} \right) X^T Y, \tag{8.5}$$

$\rho_{wb} = \frac{\lambda_w}{\lambda_w + \lambda_b}$ *and* $\mathcal{M} = \{M : M = Z(Z^T Z)^{-1} Z^T,\ Z \in \{0,1\}^{n \times Q},\ Z\mathbf{1} = \mathbf{1}\}$ *is the set of equivalence matrices.*

Proof. Using a squared loss and Kronecker's product formula, we get for a given matrix of classifier $W = (w_1, \ldots, w_K) \in \mathbb{R}^{d \times K}$,

$$\begin{aligned} L(W) &= \frac{1}{2n} \|XW - Y\|_F^2 \\ &= \frac{1}{2n} \mathbf{Tr}(W^T X^T X W) - \frac{1}{n} \mathbf{Tr}(W^T X^T Y) + \frac{1}{2n} \|Y\|_F^2 \\ &= \frac{1}{2n} \operatorname{Vect}(W)^T (\mathbf{I}_K \otimes X^T X) \operatorname{Vect}(W) - \frac{1}{n} \operatorname{Vect}(W)^T \operatorname{Vect}(X^T Y) + \frac{1}{2n} \|Y\|_F^2. \end{aligned}$$

Simplified clustering penalty computed in Lemma 8.3.1 can also be rewritten using Kronecker's product formula as

$$\min_V \Omega(W, VZ^T) = \frac{\lambda_w}{2} \operatorname{Vect}(W)^T (P \otimes \mathbf{I}_d) \operatorname{Vect}(W),$$

where $P = \mathbf{I}_K - \rho_{wb}(M - \Pi_K) - \rho_{wm}\Pi_K$. Problem (8.2) can then be simplified as

$$\begin{aligned} \min_{W,V} L(W) + \Omega(W, VZ^T) = &\min_W \frac{1}{2n} \operatorname{Vect}(W)^T (\mathbf{I}_K \otimes X^T X + P \otimes \mathbf{I}_d) \operatorname{Vect}(W) \\ &- \frac{1}{n} \operatorname{Vect}(W)^T \operatorname{Vect}(X^T Y) + \frac{1}{2n} \|Y\|_F^2 \\ = &- \frac{1}{2n} \operatorname{Vect}(X^T Y)^T \left(\mathbf{I}_K \otimes X^T X + \lambda_w n P \otimes \mathbf{I}_d \right)^{-1} \operatorname{Vect}(X^T Y) \\ &+ \frac{1}{2n} \|Y\|_F^2 \end{aligned}$$

Denote $v_1, ..., v_d \in \mathbb{R}^d, \lambda_1, ..., \lambda_d \in \mathbb{R}$ and $u_1, ..., u_K \in \mathbb{R}^K, \mu_1, ..., \mu_K \in \mathbb{R}$ the eigenvectors and corresponding eigenvalues respectively of matrices $X^T X$ and $P = (\mathbf{I}_K - \rho_{wb}(M - \Pi_K) + \rho_{wm}\Pi_K)$. The eigenvectors and corresponding eigenvalues of $\mathbf{I}_K \otimes X^T X + \lambda_w n P \otimes \mathbf{I}_d$ are $(u_i \otimes v_j)_{i \in [\![1,n]\!],\, j \in [\![1,d]\!]}$ and $(\lambda_w n \mu_i + \lambda_j)_{i \in [\![1,n]\!],\, j \in [\![1,d]\!]}$. The inversion in the expression of G is then given by

$$J^{-1} = \left(\mathbf{I}_K \otimes X^T X + \lambda_w n P \otimes \mathbf{I}_d \right)^{-1} = \sum_{i=1}^{n} \sum_{j=1}^{d} \frac{1}{\lambda_w n \mu_i + \lambda_j} u_i u_i^T \otimes v_j v_j^T.$$

We then note that the set of eigenvectors of P can be decomposed into three sets. Matrices $\mathbf{I}_K - M, M - \Pi_K$ and Π_K are indeed orthogonal projectors on orthogonal subspaces spanning the entire space. Denote by $\mathcal{I}_w, \mathcal{I}_b, \mathcal{I}_m$ the sets of eigenvectors corresponding respectively to $\mathbf{I}_K - M$, $M - \Pi_K$ and Π_K, their corresponding eigenvalues

in P can easily be computed and we obtain

$$\begin{aligned}P &= \mathbf{I}_K - M + (1-\rho_{wb})(M - \Pi_K) + (1-\rho_{wm})\Pi_K \\ &= \sum_{i\in\mathcal{I}_w} u_i u_i^T + (1-\rho_{wb}) \sum_{i\in\mathcal{I}_b} u_i u_i^T + (1-\rho_{wm}) \sum_{i\in\mathcal{I}_m} u_i u_i^T.\end{aligned}$$

This decomposition can be used for the inversion

$$\begin{aligned}J^{-1} &= \sum_{i\in\mathcal{I}_W} \sum_{j=1}^{d} \frac{1}{\lambda_w n + \lambda_j} u_i u_i^T \otimes v_j v_j^T \\ &+ \sum_{i\in\mathcal{I}_B} \sum_{j=1}^{d} \frac{1}{\lambda_w n(1-\rho_{wb}) + \lambda_j} u_i u_i^T \otimes v_j v_j^T \\ &+ \sum_{i\in\mathcal{I}_W} \sum_{j=1}^{d} \frac{1}{\lambda_w n(1-\rho_{wm}) + \lambda_j} u_i u_i^T \otimes v_j v_j^T \\ &= (\mathbf{I}_K - M) \otimes (X^T X + n\lambda_w \mathbf{I}_d)^{-1} + (M - \Pi_K) \otimes (X^T X + n\lambda_w(1-\rho_{wb})\mathbf{I}_d)^{-1} \\ &+ \Pi_K \otimes (X^T X + n\lambda_w(1-\rho_{wm})\mathbf{I}_d)^{-1}.\end{aligned}$$

Finally clustered multitask problem can be simplified using properties of the Kronecker product

$$\begin{aligned}A =& \min_{W,V} L(W) + \Omega(W, VZ^T) \\ =& -\frac{1}{2n} \operatorname{Vect}(X^T Y)^T \left((\mathbf{I}_K - M) \otimes (X^T X + n\lambda_w \mathbf{I}_d)^{-1}\right) \operatorname{Vect}(X^T Y) \\ &- \frac{1}{2n} \operatorname{Vect}(X^T Y)^T \left((M - \Pi_K) \otimes (X^T X + n\lambda_w(1-\rho_{wb})\mathbf{I}_d)^{-1}\right) \operatorname{Vect}(X^T Y) \\ &- \frac{1}{2n} \operatorname{Vect}(X^T Y)^T \left(\Pi_K \otimes (X^T X + n\lambda_w(1-\rho_{wm})\mathbf{I}_d)^{-1}\right) \operatorname{Vect}(X^T Y) + \frac{1}{2n}\|Y\|_F^2 \\ =& -\frac{1}{2n} \mathbf{Tr}(Y^T X (X^T X + n\lambda_w \mathbf{I}_d)^{-1} X^T Y (\mathbf{I}_K - M)) \\ &- \frac{1}{2n} \mathbf{Tr}(Y^T X (X^T X + n\lambda_w(1-\rho_{wb})\mathbf{I}_d)^{-1} X^T Y (M - \Pi_K)) \\ &- \frac{1}{2n} \mathbf{Tr}(Y^T X (X^T X + n\lambda_w(1-\rho_{wm})\mathbf{I}_d)^{-1} X^T Y \Pi_K) + \frac{1}{2n}\|Y\|_F^2.\end{aligned}$$

The remaining variable of the problem is the normalized equivalence matrix M. By ignoring the constant terms we get the claimed formulation. ■

Clustered multitask problem reduces therefore to a linear problem in normalized equivalence matrices M of partitions of the K classes. Though the feasible set is still non-convex, minimization of this linear function on the set of normalized equivalence matrices reduces to a k-means problem that can be approximated efficiently.

8.3.2 Resolution by k-means

As $0 \leq \rho_{wb} \leq 1$, we get that A defined in (8.5) is semi-definite negative and denote $U \in \mathbb{R}^{K \times K}$ a squared root of $-A$ such that $-A = UU^T$. Clustered multitask problem (8.4) reads then

$$\begin{aligned}\operatorname*{argmin}_{S \in \mathcal{M}} \mathbf{Tr}(S^T A) &= \operatorname*{argmin}_{S \in \mathcal{M}} -\mathbf{Tr}(SUU^T) \\ &= \operatorname*{argmin}_{S \in \mathcal{M}} \mathbf{Tr}((\mathbf{I} - S)UU^T)) \\ &\overset{\vartheta}{=} \operatorname*{argmin}_{S \in \mathcal{M}} \|U - SU\|_F^2. \end{aligned} \tag{8.6}$$

In ϑ, we used that normalized equivalence matrices are orthogonal projectors, so $I - S$ is also an orthogonal projector. Now we observe that (8.6) is a k-means problem as presented in Section 5.2.2. Therefore solving a k-means problem on the rows of the squared root matrix U of $-\nabla f(M)$ offers a solution to the clustered multi-task problem with squared loss. Careful initialization as made in k-means++ [Arthur and Vassilvitskii, 2007] offers logarithmic approximations to the problem.

Resolution of the clustered-multitask problem with squared loss by k-means is summarized in Algorithm 11. We denote $Z = \text{k-means}(U, Q)$ the assignment matrix found by a dedicated algorithm that clusters the rows of U into Q groups. Once an equivalence matrix found for the problem, classifiers stacked in $W = (w_1, \ldots, w_k)$ are given in (8.7) using computations made in Proposition 8.3.2 and Kronecker's formula. Finally ρ_{wb}, ρ_{wm} are defined in Lemma 8.3.1.

Algorithm 11 Clustered multitask resolution by k-means

Inputs: Data (X, Y), desired number of groups Q, regularization parameters $\lambda_w, \lambda_b, \lambda_m$.
Compute squared root U of

$$A = Y^T X \left((X^T X + n\lambda_w \mathbf{I}_d)^{-1} - (X^T X + n\lambda_w (1 - \rho_{wb}) \mathbf{I}_d)^{-1} \right) X^T Y$$

Get

$$Z = \text{k-means}(U, Q), \quad M = Z(Z^T Z)^\dagger Z^T$$

Output:

$$\begin{aligned} W = \quad & (X^T X + n\lambda_w \mathbf{I}_d)^{-1} X^T Y (\mathbf{I}_K - M) \\ & + (X^T X + n\lambda_w (1 - \rho_{wb}) \mathbf{I}_d)^{-1} X^T Y (M - \Pi_K) \\ & + (X^T X + n\lambda_w (1 - \rho_{wm}) \mathbf{I}_d)^{-1} X^T Y \Pi_K \end{aligned} \tag{8.7}$$

Overall this solution requires a k-means++ initialization that costs $O(Q^2 K^2)$ operations and the resolution of the k-means problem that standard alternating minimization approximates at a cost of $O(TQK^2)$, where T is the number of alternating steps, which is generally small. The main drawback of this solution is the computation of A in (8.5) that requires the inversion of matrices of size $d \times d$ and the computation

of its squared root. This burdens its implementation for big data sets. However it can give a good initialization for more elaborated approaches that use different losses than the squared loss.

8.4 Numerical experiments

We perform classification of documents in topics. We used the publicly available 20NewsGroup dataset which contains 2800 documents that are classified in 20 topics. Some of the topics are very closely related to each other (*e.g.* `pc.hardware` and `mac.hardware`), while others are highly unrelated (*e.g.* `misc.forsale` and `soc.religion.christian`). Using this prior, we try to benefit from the classification of some topics with the use of similar topics. We use a dictionary of 5000 words selected by TF-IDF coefficients and take the word frequencies as covariates for each document.

In Table 8.1, we compare our approach to other classical regularizations such as the Frobenius norm and the trace norm, as implemented by Ciliberto et al. [2015], using either a ridge or a logistic loss (Log). The algorithm proposed by Jacob et al. [2009] was too slow on this large dataset to compare with. We initialize it by the solution given by the logistic loss. All algorithms were 5-fold cross-validated on 80% of the data then tested on the remaining 20%. The number of clusters was set to 5, as suggested by the names of newsgroups. Figures after the $\pm$ sign correspond to one standard deviation when varying the training and test sets. At first glance, our solutions appear to offer better generalization performance, however the groups of tasks found by our method do not have meaning and best regularization parameters are the ones that do not tend to clustered tasks.

Frobenius Log	Trace Ridge	**CG Ridge**	**PG Log**
5.8±2.0	4.3±0.8	4.9±0.8	**2.6**±1.5

Table 8.1 – 100 $\times$ mean absolute errors for predicting topics on 20NewsGroup dataset, comparing classical regularizers (Frobenius and Trace) with our algorithms. PG refers to projected gradient, CG refers to conditional gradient.

8.5 Conclusion

We applied our framework of learning problems with partitions to cluster tasks. Once more a projected gradient scheme can be applied to offer scalable resolution for any loss. However framework of Jacob et al. [2009] requires to tune at least 3 parameters which burdens its application. Moreover regularization of the variance between clusters is generally taken positive to develop convex relaxations. Further experiments that both enforce low variance intra-clusters and high variance inter-clusters may better benefit from the clustering approach. Notice that our computations for

squared loss revealed that this framework can be reduced to a k-means problem for which several approaches exist that may refine the framework.

Appendix A

Classical algorithms implementation

We present here the classical algorithms for convex optimization that we restart. We present their general form to solve composite optimization problems of the form

$$\text{minimize} \quad f(x) = \phi(x) + g(x) \tag{Composite}$$

where ϕ, g are convex functions and g is assumed simple in a sense that will be clarified later.

Algorithms used to solve this problem crucially depend on the smoothness assumption on the (sub)gradients of ϕ. We assume here that they are Hölder continuous with respect to a given norm $\|\cdot\|$ on a set J, i.e. that there exists $s \in [0,1]$ and $L > 0$ such that

$$\|\nabla\phi(x) - \nabla\phi(y)\|_* \leq \|x - y\|^{s-1}, \quad \text{for every } x, y \in J, \tag{Generic Smoothness}$$

where $\|\cdot\|$ is the dual norm of $\|\cdot\|$ and $\nabla\phi(x)$ denotes any subgradient of ϕ at x. To exploit the smoothness of ϕ with respect to a generic norm, we assume that we have access to a prox function h with $\mathbf{dom}(f) \subset \mathbf{dom}(h)$, strongly convex with respect to the norm $\|\cdot\|$ with convexity parameter equal to one, i.e.

$$h(y) \geq h(x) + \nabla h(x)^T(y - x) + \frac{1}{2}\|x - y\|^2, \quad \text{for any } x, y \in \mathbf{dom}(h).$$

We define the Bregman divergence associated to h as, for given $x, y \in \mathbf{dom}(h)$,

$$D_h(y; x) = h(y) - h(x) - \nabla h(x)^T(y - x)$$

such that we naturally have $D_h(y; x) \geq \frac{1}{2}\|x - y\|^2$. For $h(x) = \frac{1}{2}\|x\|_2^2$, we get $D_h(y; x) = \frac{1}{2}\|x - y\|_2^2$, so we retrieve the Euclidean setting. The assumption that g is "simple" can now be stated formally. Given $x, y \in \mathbf{dom}(f)$ and $\lambda \geq 0$ we assume that we can solve

$$\min_z y^T z + g(z) + \lambda D_h(z; x)$$

either in a closed form or by some cheap computational procedure.

A.1 Universal fast gradient method

An optimal algorithm to solve the (Composite) problem is then the universal fast gradient method [Nesterov, 2015]. It is detailed in Algorithm 12. Given a target accuracy ε, it starts at a point x_0 and outputs after t iterations a point $x \triangleq \mathcal{U}(x_0, \varepsilon, t)$, such that

$$f(x) - f^* \leq \frac{\varepsilon}{2} + \frac{c_s L^{\frac{2}{s}} D_h(x_0, X^*)}{\varepsilon^{\frac{2}{s}} t^{\frac{2\rho}{s}}} \frac{\varepsilon}{2},$$

where $D_h(x; X^*) = \min_{x^* \in X^*} D_h(x; x^*)$ is the Bregman distance from x to the set of minimizers and $c_s = 2^{\frac{5s-2}{s}}$ that we bounded as $c_s \leq c = 16$ for $s \in [1, 2]$.

$$\rho \triangleq \frac{3s-2}{2}$$

is the optimal rate of convergence for s-smooth functions. In the Euclidean setting, $h = \frac{1}{2}\|x\|_2^2$, $D_h(y; x) = \frac{1}{2}\|x - y\|^2$, such that we get the bound given in (2.17).

The method does not need to know the smoothness parameters (s, L), but the target accuracy ε is used to parametrize the algorithm. The universal fast gradient method requires an estimate L_0 of the smoothness parameter L to start a line search on L. This line search is proven to increase the complexity of the algorithm by at most a constant factor plus a logarithmic term and ensures that the overall complexity does not depend on L_0 but on L. In our restart schemes we use a first estimate L_0 when running the algorithm for the first time and we use the last estimate found by the algorithm when restarting it.

Finally if $X^* \neq \emptyset$, the universal fast gradient method produces a convergent sequence of iterates. Therefore if the Łojasiewicz inequality is satisfied on a compact set K, it will be valid for all our iterates after perhaps reducing μ.

A.2 Accelerated gradient method

The accelerated gradient method is a special instance of the universal fast gradient method when the function ϕ is known to be smooth (i.e. satisfies (Generic Smoothness) with $s = 2$). In that case the optimal ε to run the Universal Fast Gradient method is 0 (otherwise it depends on the parameters of the function). Given an initial point x_0, accelerated gradient method outputs, after t iterations, a point $x \triangleq \mathcal{A}(x_0, t) = \mathcal{U}(x_0, 0, t)$ such that

$$f(y) - f^* \leq \frac{cL}{t^2} D_h(x_0, X^*),$$

where $D_h(x; X^*) = \min_{x^* \in X^*} D_h(x; x^*)$ is the Bregman distance from x to the set of minimizers and $c = 8$. In the Euclidean setting, $D_h(y; x) = \frac{1}{2}\|x - y\|^2$, such that we get the bound given in (2.3). Here again smoothness parameter L is found by a backtracking line search such that one only needs a first estimate of its value.

Algorithm 12 Universal fast gradient method

Inputs : x_0, L_0, ε
Initialize : $y_0 := x_0$, $A_0 := 0$, $\hat{L} := L_0$
for $t = 0, \ldots, T$ **do**

$$z_t := \arg\min_z \sum_{i=1}^{t} a_i \nabla\phi(x_i)^T z + A_t g(z) + D_h(z; x_0)$$

repeat
Find $a \geq 0$, such that

$$a^2 = \frac{1}{\hat{L}}(A_t + a)$$

Choose

$$\begin{aligned} \tau &:= \frac{a}{A_t + a} \\ x &:= \tau z_t + (1-\tau) y_t \\ \hat{x} &:= \arg\min_z \; a\nabla\phi(x)^T z + a\psi(z) + D_h(z; z_t) \\ y &:= \tau\hat{x} + (1-\tau) y_t \end{aligned}$$

if $\phi(y) \geq \phi(x) + \langle \nabla\phi(x), y - x\rangle + \frac{\hat{L}}{2}\|y-x\|_2^2 + \frac{\tau\varepsilon}{2}$ **then** $\hat{L} := 2\hat{L}$ **end if**
until $\phi(y) \leq \phi(x) + \langle \nabla\phi(x), y - x\rangle + \frac{\hat{L}}{2}\|y-x\|_2^2 + \frac{\tau\varepsilon}{2}$
Set

$$\begin{gathered} x_{t+1} := x, \quad y_{t+1} := y, \quad a_{t+1} := a, \\ A_{t+1} := A_t + a_{t+1}, \quad \hat{L} := \hat{L}/2, \end{gathered}$$

end for
Output : $x = y_T$

A.3 Gradient descent method

We recall in Algorithm 13 the simple gradient descent method when the function ϕ is smooth with constant L. I can also be found in Nesterov [2015]. It starts at a point x_0 and outputs iterates $x_t = \mathcal{G}(x_0, t)$ such that

$$f(x_t) - f^* \leq \frac{cL}{t} D_h(x_0, X^*),$$

where $c = 2$ and $D_h(x; X^*) = \min_{x^* \in X^*} D_h(x; x^*)$ is the Bregman distance from x to the set of minimizers. In the Euclidean setting, $D_h(y; x) = \frac{1}{2}\|x - y\|^2$, such that we get the bound in (2.15). Once again it performs a line search on the smoothness parameter L such that L_0 can be chosen arbitrarily.

Algorithm 13 Gradient descent method

Inputs : x_0, L_0
Initialize : $\hat{L} := L_0$
for $t = 0, \ldots$ **do**
  repeat
    $x := \arg\min_z \nabla\phi(x)^T z + g(z) + \hat{L} D_h(z; x)$
    if $\phi(x) \geq \phi(x_t) + \langle \nabla\phi(x_t), x - x_t \rangle + \frac{\hat{L}}{2}\|x - x_t\|_2^2$ **then** $\hat{L} = 2\hat{L}$ **end if**
  until $\phi(x) \leq \phi(x_t) + \langle \nabla\phi(x_t), x - x_t \rangle + \frac{\hat{L}}{2}\|x - x_t\|_2^2$
  Set

$$x_{t+1} := x, \quad \hat{L} := \hat{L}/2$$

end for

Bibliography

Agarwal, A., Negahban, S. and Wainwright, M. J. [2010], Fast global convergence rates of gradient methods for high-dimensional statistical recovery, *in* 'Advances in Neural Information Processing Systems', pp. 37–45.

Amelunxen, D. and Lotz, M. [2014], 'Gordon's inequality and condition numbers in conic optimization', *arXiv preprint arXiv:1408.3016* .

Amelunxen, D., Lotz, M., McCoy, M. B. and Tropp, J. A. [2014], 'Living on the edge: Phase transitions in convex programs with random data', *Information and Inference* p. iau005.

Amini, O., Mazoit, F., Nisse, N. and Thomassé, S. [2009], 'Submodular partition functions', *Discrete Mathematics* **309**(20), 6000–6008.

Argyriou, A., Evgeniou, T. and Pontil, M. [2008], 'Convex multi-task feature learning', *Machine Learning* **73**(3), 243–272.

Arthur, D. and Vassilvitskii, S. [2007], k-means++: The advantages of careful seeding, *in* 'Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms', Society for Industrial and Applied Mathematics, pp. 1027–1035.

Asif, M. S. and Romberg, J. [2014], 'Sparse recovery of streaming signals using l1-homotopy', *Signal Processing, IEEE Transactions on* **62**(16), 4209–4223.

Asuncion, A. and Newman, D. [2007], 'Uci machine learning repository'.

Attouch, H., Bolte, J., Redont, P. and Soubeyran, A. [2010], 'Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the kurdyka-łojasiewicz inequality', *Mathematics of Operations Research* **35**(2), 438–457.

Attouch, H., Bolte, J. and Svaiter, B. F. [2013], 'Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods', *Mathematical Programming* **137**(1-2), 91–129.

Auslender, A. and Crouzeix, J.-P. [1988], 'Global regularity theorems', *Mathematics of Operations Research* **13**(2), 243–253.

Auslender, A. and Teboulle, M. [2006], 'Interior gradient and proximal methods for convex and conic optimization', *SIAM Journal on Optimization* **16**(3), 697–725.

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

Bach, F. R. and Harchaoui, Z. [2008], Diffrac: a discriminative and flexible framework for clustering, *in* 'Advances in Neural Information Processing Systems', pp. 49–56.

Bach, F. et al. [2013], 'Learning with submodular functions: A convex optimization perspective', *Foundations and Trends® in Machine Learning* **6**(2-3), 145–373.

Bagirov, A. M., Ugon, J. and Mirzayeva, H. [2013], 'Nonsmooth nonconvex optimization approach to clusterwise linear regression problems', *European Journal of Operational Research* **229**(1), 132–142.

Balding, D. J. [2006], 'A tutorial on statistical methods for population association studies', *Nature reviews. Genetics* **7**(10), 781.

Banerjee, A., Merugu, S., Dhillon, I. S. and Ghosh, J. [2005], 'Clustering with bregman divergences', *Journal of machine learning research* **6**(Oct), 1705–1749.

Bauschke, H. H., Bolte, J. and Teboulle, M. [2016], 'A descent lemma beyond lipschitz gradient continuity: first-order methods revisited and applications', *Mathematics of Operations Research* **42**(2), 330–348.

Bauschke, H. H. and Combettes, P. L. [2011], *Convex analysis and monotone operator theory in Hilbert spaces*, Vol. 408, Springer.

Beck, A. and Shtern, S. [2015], 'Linearly convergent away-step conditional gradient for non-strongly convex functions', *Mathematical Programming* pp. 1–27.

Beck, A. and Teboulle, M. [2003], 'Mirror descent and nonlinear projected subgradient methods for convex optimization', *Operations Research Letters* **31**(3), 167–175.

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

Becker, S., Bobin, J. and Candès, E. J. [2011], 'Nesta: A fast and accurate first-order method for sparse recovery', *SIAM Journal on Imaging Sciences* **4**(1), 1–39.

Becker, S. R., Candès, E. J. and Grant, M. C. [2011], 'Templates for convex cone problems with applications to sparse signal recovery', *Mathematical Programming Computation* **3**(3), 165–218.

Bellman, R. [1973], 'A note on cluster analysis and dynamic programming', *Mathematical Biosciences* **18**(3), 311–312.

Belloni, A. and Freund, R. M. [2009], 'A geometric analysis of renegar's condition number, and its interplay with conic curvature', *Mathematical programming* **119**(1), 95–107.

Belloni, A., Freund, R. M. and Vempala, S. [2009], 'An efficient rescaled perceptron algorithm for conic systems', *Mathematics of Operations Research* **34**(3), 621–641.

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

Bickel, P. J., Ritov, Y. and Tsybakov, A. B. [2009], 'Simultaneous analysis of lasso and dantzig selector', *The Annals of Statistics* **37**(4), 1705–1732.

Bierstone, E. and Milman, P. D. [1988], 'Semianalytic and subanalytic sets', *Publications Mathématiques de l'IHÉS* **67**, 5–42.

Blanchet, A. and Bolte, J. [2016], 'A family of functional inequalities: lojasiewicz inequalities and displacement convex functions', *arXiv preprint arXiv:1612.02619* .

Blumensath, T. and Davies, M. E. [2009], 'Iterative hard thresholding for compressed sensing', *Applied and Computational Harmonic Analysis* **27**(3), 265–274.

Bolte, J., Daniilidis, A. and Lewis, A. [2007], 'The łojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems', *SIAM Journal on Optimization* **17**(4), 1205–1223.

Bolte, J., Nguyen, T. P., Peypouquet, J. and Suter, B. W. [2015], 'From error bounds to the complexity of first-order descent methods for convex functions', *Mathematical Programming* pp. 1–37.

Bolte, J., Sabach, S. and Teboulle, M. [2014], 'Proximal alternating linearized minimization for nonconvex and nonsmooth problems', *Mathematical Programming* **146**(1-2), 459–494.

Bondell, H. D. and Reich, B. J. [2008], 'Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with oscar', *Biometrics* **64**(1), 115–123.

Borwein, J. and Lewis, A. S. [2010], *Convex analysis and nonlinear optimization: theory and examples*, Springer Science & Business Media.

Boumal, N. [2016], 'Nonconvex phase synchronization', *SIAM Journal on Optimization* **26**(4), 2355–2377.

Boumal, N., Mishra, B., Absil, P.-A. and Sepulchre, R. [2014], 'Manopt, a Matlab toolbox for optimization on manifolds', *Journal of Machine Learning Research* **15**, 1455–1459.
URL: *http://www.manopt.org*

Boyd, S. and Vandenberghe, L. [2004], *Convex optimization*, Cambridge University Press.

Broyden, C. G. [1970], 'The convergence of a class of double-rank minimization algorithms: 2. the new algorithm', *IMA journal of applied mathematics* **6**(3), 222–231.

Bühlmann, P., Rütimann, P., van de Geer, S. and Zhang, C.-H. [2013], 'Correlated variables in regression: clustering and sparse estimation', *Journal of Statistical Planning and Inference* **143**(11), 1835–1858.

Burke, J. and Deng, S. [2002], 'Weak sharp minima revisited part i: basic theory', *Control and Cybernetics* **31**, 439–469.

Burke, J. and Ferris, M. C. [1993], 'Weak sharp minima in mathematical programming', *SIAM Journal on Control and Optimization* **31**(5), 1340–1359.

Candès, E. J., Romberg, J. and Tao, T. [2006], 'Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information', *IEEE Transactions on information theory* **52**(2), 489–509.

Candès, E. J. and Tao, T. [2005], 'Decoding by linear programming', *IEEE Transactions on Information Theory* **51**(12), 4203–4215.

Candes, E. J. and Tao, T. [2006], 'Near-optimal signal recovery from random projections: Universal encoding strategies?', *IEEE transactions on information theory* **52**(12), 5406–5425.

Candès, E. J. and Wakin, M. B. [2008], 'An introduction to compressive sampling', *IEEE signal processing magazine* **25**(2), 21–30.

Candes, E. and Tao, T. [2007], 'The dantzig selector: Statistical estimation when p is much larger than n', *The Annals of Statistics* pp. 2313–2351.

Cauchy, A. [1847], 'Méthode générale pour la résolution des systemes d'équations simultanées', *Comp. Rend. Sci. Paris* **25**(1847), 536–538.

Chambolle, A., De Vore, R. A., Lee, N.-Y. and Lucier, B. J. [1998], 'Nonlinear wavelet image processing: variational problems, compression, and noise removal through wavelet shrinkage', *IEEE Transactions on Image Processing* **7**(3), 319–335.

Chandrasekaran, V. and Jordan, M. I. [2013], 'Computational and statistical tradeoffs via convex relaxation', *Proceedings of the National Academy of Sciences* **110**(13), 1181–1190.

Chandrasekaran, V., Recht, B., Parrilo, P. and Willsky, A. [2012], 'The convex geometry of linear inverse problems', *Foundations of Computational Mathematics* **12**(6), 805–849.

Chen, S. S., Donoho, D. L. and Saunders, M. A. [2001], 'Atomic decomposition by basis pursuit', *SIAM review* **43**(1), 129–159.

Ciliberto, C., Mroueh, Y., Poggio, T. and Rosasco, L. [2015], Convex learning of multiple tasks and their structure, *in* 'Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015', pp. 1548–1557.

Cohen, A., Dahmen, W. and DeVore, R. [2009], 'Compressed sensing and best k-term approximation', *Journal of the American mathematical society* **22**(1), 211–231.

d'Aspremont, A., Guzmán, C. and Jaggi, M. [2013], 'An optimal affine invariant smooth minimization algorithm', *arXiv preprint arXiv:1301.0465* .

Dedieu, J. P. [1992], 'Penalty functions in subanalytic optimization', *Optimization* **26**(1-2), 27–32.

Dempster, A. P., Laird, N. M. and Rubin, D. B. [1977], 'Maximum likelihood from incomplete data via the em algorithm', *Journal of the royal statistical society. Series B (methodological)* pp. 1–38.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L. [2009], ImageNet: A Large-Scale Hierarchical Image Database, *in* 'CVPR09'.

Deshpande, Y., Montanari, A. and Richard, E. [2014], Cone-constrained principal component analysis, *in* 'Advances in Neural Information Processing Systems', pp. 2717–2725.

Devolder, O., Glineur, F. and Nesterov, Y. [2014], 'First-order methods of smooth convex optimization with inexact oracle', *Mathematical Programming* **146**(1-2), 37–75.

Dhillon, I. S., Guan, Y. and Kulis, B. [2004], Kernel k-means: spectral clustering and normalized cuts, *in* 'Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 551–556.

Donoho, D. and Huo, X. [2001], 'Uncertainty principles and ideal atomic decomposition', *IEEE Transactions on Information Theory* **47**(7), 2845–2862.

Donoho, D. L. [2006], 'For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution', *Communications on pure and applied mathematics* **59**(6), 797–829.

Donoho, D. L. and Tanner, J. [2005], 'Sparse nonnegative solutions of underdetermined linear equations by linear programming', *Proc. of the National Academy of Sciences* **102**(27), 9446–9451.

Donoho, D. L. and Tsaig, Y. [2008], 'Fast solution of ℓ_1-norm minimization problems when the solution may be sparse-norm minimization problems when the solution may be sparse', *Information Theory, IEEE Transactions on* **54**(11), 4789–4812.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R. et al. [2004], 'Least angle regression', *The Annals of statistics* **32**(2), 407–499.

Epelman, M. and Freund, R. M. [2000], 'Condition number complexity of an elementary algorithm for computing a reliable solution of a conic linear system', *Mathematical Programming* **88**(3), 451–485.

Evgeniou, T. and Pontil, M. [2004], Regularized multi–task learning, *in* 'Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 109–117.

Fercoq, O., Gramfort, A. and Salmon, J. [2015], Mind the duality gap: safer rules for the lasso, *in* 'International Conference on Machine Learning', pp. 333–342.

Fercoq, O. and Qu, Z. [2016], 'Restarting accelerated gradient methods with a rough strong convexity estimate', *arXiv preprint arXiv:1609.07358* .

Fercoq, O. and Qu, Z. [2017], 'Adaptive restart of accelerated gradient methods under local quadratic growth condition', *arXiv preprint arXiv:1709.02300* .

Fercoq, O. and Richtárik, P. [2015], 'Accelerated, parallel, and proximal coordinate descent', *SIAM Journal on Optimization* **25**(4), 1997–2023.

Feuer, A. and Nemirovski, A. [2003], 'On sparse representation in pairs of bases', *IEEE Transactions on Information Theory* **49**(6), 1579–1581.

Fletcher, R. [1970], 'A new approach to variable metric algorithms', *The computer journal* **13**(3), 317–322.

Frank, M. and Wolfe, P. [1956], 'An algorithm for quadratic programming', *Naval research logistics quarterly* **3**(1-2), 95–110.

Frankel, P., Garrigos, G. and Peypouquet, J. [2015], 'Splitting methods with variable metric for kurdyka–łojasiewicz functions and general convergence rates', *Journal of Optimization Theory and Applications* **165**(3), 874–900.

Freund, R. M. and Lu, H. [2015], 'New computational guarantees for solving convex optimization problems with first order methods, via a function growth condition measure', *arXiv preprint arXiv:1511.02974* .

Freund, R. M. and Vera, J. R. [1999*a*], 'Condition-based complexity of convex optimization in conic linear form via the ellipsoid algorithm', *SIAM Journal on Optimization* **10**(1), 155–176.

Freund, R. M. and Vera, J. R. [1999*b*], 'Some characterizations and properties of the "distance to ill-posedness" and the condition measure of a conic linear system', *Mathematical Programming* **86**(2), 225–260.

Freund, R. M. and Vera, J. R. [2003], 'On the complexity of computing estimates of condition measures of a conic linear system', *Mathematics of Operations Research* **28**(4), 625–648.

Fujishige, S. [2005], *Submodular functions and optimization*, Vol. 58, Elsevier.

Gan, G., Ma, C. and Wu, J. [2007], *Data clustering: theory, algorithms, and applications*, SIAM.

Gautschi, W. [2011], *Numerical analysis*, Springer Science & Business Media.

Gilpin, A., Pena, J. and Sandholm, T. [2012], 'First-order algorithm with $\mathcal{O}(\log 1/\epsilon)$ convergence for ϵ-equilibrium in two-person zero-sum games', *Mathematical programming* **133**(1-2), 279–298.

Giselsson, P. and Boyd, S. [2014], Monotonicity and restart in fast gradient methods, *in* '53rd IEEE Conference on Decision and Control', IEEE, pp. 5058–5063.

Goldfarb, D. [1970], 'A family of variable-metric methods derived by variational means', *Mathematics of computation* **24**(109), 23–26.

Grant, M., Boyd, S. and Ye, Y. [2001], 'CVX: Matlab software for disciplined convex programming'.

Graubard, B. I. and Korn, E. L. [1994], 'Regression analysis with clustered data', *Statistics in medicine* **13**(5-7), 509–522.

Gupta, V., Lehal, G. S. et al. [2009], 'A survey of text mining techniques and applications', *Journal of emerging technologies in web intelligence* **1**(1), 60–76.

Guyon, I. and Elisseeff, A. [2003], 'An introduction to variable and feature selection', *Journal of machine learning research* **3**(Mar), 1157–1182.

Guzman-Rivera, A., Kohli, P., Batra, D. and Rutenbar, R. [2014], Efficiently enforcing diversity in multi-output structured prediction, *in* 'Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics', pp. 284–292.

Harchaoui, Z. [2013], 'Large-scale learning for image classification', http://www.di.ens.fr/willow/events/cvml2013/materials/slides/thursday/harch_cvml13.pdf.

Hastie, T., Tibshirani, R., Botstein, D. and Brown, P. [2001], 'Supervised harvesting of expression trees', *Genome Biology* **2**(1), research0003–1.

Hastie, T., Tibshirani, R. and Friedman, J. [2008], *The elements of statistical learning: data mining, inference and prediction*, 2 edn, Springer.

Hestenes, M. R. and Stiefel, E. [1952], 'Methods of conjugate gradients for solving linear systems", *Journal of Research of the National Bureau of Standards* **49**(6).

Hocking, T. D., Joulin, A., Bach, F. and Vert, J.-P. [2011], Clusterpath an algorithm for clustering using convex fusion penalties, *in* '28th international conference on machine learning', p. 1.

Hoffman, A. J. [1952], 'On approximate solutions of systems of linear inequalities', *Journal of Research of the National Bureau of Standards* **49**(4).

Horn, R. A. and Johnson, C. R. [1990], *Matrix analysis*, Cambridge University Press, Cambridge. Corrected reprint of the 1985 original.

Ito, M. and Fukuda, M. [2016], 'A family of subgradient-based methods for convex optimization problems in a unifying framework', *Optimization Methods and Software* **31**(5), 952–982.

Jacob, L., Obozinski, G. and Vert, J.-P. [2009], Group lasso with overlap and graph lasso, *in* 'Proceedings of the 26th annual international conference on machine learning', ACM, pp. 433–440.

Jaggi, M. [2013], Revisiting frank-wolfe: Projection-free sparse convex optimization, *in* 'Proceedings of the 30th International Conference on Machine Learning (ICML-13)', pp. 427–435.

Jain, P., Tewari, A. and Kar, P. [2014], On iterative hard thresholding methods for high-dimensional m-estimation, *in* 'Advances in Neural Information Processing Systems', pp. 685–693.

Jordan, M. I. [1994], 'Hierarchical mixtures of experts and the em algorithm', *Neural Computation* **6**, 181–214.

Journée, M., Nesterov, Y., Richtárik, P. and Sepulchre, R. [2010], 'Generalized power method for sparse principal component analysis', *Journal of Machine Learning Research* **11**(Feb), 517–553.

Juditski, A. and Nesterov, Y. [2014], 'Primal-dual subgradient methods for minimizing uniformly convex functions', *arXiv preprint arXiv:1401.1792* .

Juditsky, A., Karzan, F. K. and Nemirovski, A. [2014], 'On a unified view of nullspace-type conditions for recoveries associated with general sparsity structures', *Linear Algebra and its Applications* **441**, 124–151.

Karimi, H., Nutini, J. and Schmidt, M. [2016], Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition, *in* 'Joint European Conference on Machine Learning and Knowledge Discovery in Databases', Springer, pp. 795–811.

Kashin, B. S. and Temlyakov, V. N. [2007], 'A remark on compressed sensing', *Mathematical notes* **82**(5), 748–755.

Krichene, W., Bayen, A. and Bartlett, P. L. [2015], Accelerated mirror descent in continuous and discrete time, *in* 'Advances in neural information processing systems', pp. 2845–2853.

Kumar, A. and Daume, H. [2012], Learning task grouping and overlap in multi-task learning, *in* 'Proceedings of the 29th International Conference on Machine Learning (ICML-12)', pp. 1383–1390.

Kurdyka, K. [1998], On gradients of functions definable in o-minimal structures, *in* 'Annales de l'institut Fourier', Vol. 48, Chartres: L'Institut, 1950-, pp. 769–784.

Lacoste-Julien, S., Schmidt, M. and Bach, F. [2012], 'A simpler approach to obtaining an o (1/t) convergence rate for the projected stochastic subgradient method', *arXiv preprint arXiv:1212.2002* .

Lee, Y. T. and Sidford, A. [2013], Efficient accelerated coordinate descent methods and faster algorithms for solving linear systems, *in* 'Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on', IEEE, pp. 147–156.

Lessard, L., Recht, B. and Packard, A. [2016], 'Analysis and design of optimization algorithms via integral quadratic constraints', *SIAM Journal on Optimization* **26**(1), 57–95.

Li, G. [2013], 'Global error bounds for piecewise convex polynomials', *Mathematical Programming* pp. 1–28.

Li, G., Mordukhovich, B., Nghia, T. and Pham, T. [2015], 'Error bounds for parametric polynomial systems with applications to higher-order stability analysis and convergence rates', *Mathematical Programming* pp. 1–34.

Li, G., Mordukhovich, B. S. and Pham, T. [2015], 'New fractional error bounds for polynomial systems with applications to hölderian stability in optimization and spectral theory of tensors', *Mathematical Programming* **153**(2), 333–362.

Lin, Q. and Xiao, L. [2014], An adaptive accelerated proximal gradient method and its homotopy continuation for sparse optimization., *in* 'ICML', pp. 73–81.

Liu, J. and Wright, S. J. [2015], 'Asynchronous stochastic coordinate descent: Parallelism and convergence properties', *SIAM Journal on Optimization* **25**(1), 351–376.

Łojasiewicz, S. [1958], 'Division d'une distribution par une fonction analytique de variables rélles', *Comptes rendus de l'Acad'émie des Sciences de Paris* **246**, 683–686.

Łojasiewicz, S. [1961], 'Sur le probleme de la division'.

Łojasiewicz, S. [1963], 'Une propriété topologique des sous-ensembles analytiques réels', *Les équations aux dérivées partielles* pp. 87–89.

Łojasiewicz, S. [1965], *Ensembles semi-analytiques*, IHES.

Łojasiewicz, S. [1993], 'Sur la géométrie semi-et sous-analytique', *Annales de l'institut Fourier* **43**(5), 1575–1595.

Lu, H., Freund, R. M. and Nesterov, Y. [2016], 'Relatively-smooth convex optimization by first-order methods, and applications', *arXiv preprint arXiv:1610.05708* .

Luo, X.-D. and Luo, Z.-Q. [1994], 'Extension of hoffman's error bound to polynomial systems', *SIAM Journal on Optimization* **4**(2), 383–392.

Luo, Z.-Q. and Pang, J.-S. [1994], 'Error bounds for analytic systems and their applications', *Mathematical Programming* **67**(1-3), 1–28.

Luo, Z.-Q. and Sturm, J. F. [2000], Error bounds for quadratic systems, *in* 'High performance optimization', Springer, pp. 383–404.

Luss, R. and Teboulle, M. [2013], 'Conditional gradient algorithmsfor rank-one matrix approximations with a sparsity constraint', *SIAM Review* **55**(1), 65–98.

MacQueen, J. et al. [1967], Some methods for classification and analysis of multivariate observations, *in* 'Proceedings of the fifth Berkeley symposium on mathematical statistics and probability', number 14 *in* '1', Oakland, CA, USA., pp. 281–297.

Mahajan, M., Nimbhorkar, P. and Varadarajan, K. [2012], 'The planar k-means problem is np-hard', *Theoretical Computer Science* **442**, 13–21.

Mangasarian, O. L. [1985], 'A condition number for differentiable convex inequalities', *Mathematics of Operations Research* **10**(2), 175–179.

Maurer, A., Pontil, M. and Romera-Paredes, B. [2016], 'The benefit of multitask representation learning', *The Journal of Machine Learning Research* **17**(1), 2853–2884.

Natarajan, B. K. [1995], 'Sparse approximate solutions to linear systems', *SIAM journal on computing* **24**(2), 227–234.

Necoara, I., Nesterov, Y. and Glineur, F. [2015], 'Linear convergence of first order methods for non-strongly convex optimization', *arXiv preprint arXiv:1504.06298* .

Negahban, S., Yu, B., Wainwright, M. J. and Ravikumar, P. K. [2009], A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers, *in* 'Advances in Neural Information Processing Systems', pp. 1348–1356.

Nemirovski, A. [2004], 'Prox-method with rate of convergence o (1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems', *SIAM Journal on Optimization* **15**(1), 229–251.

Nemirovskii, A. and Nesterov, Y. [1985], 'Optimal methods of smooth convex minimization', *USSR Computational Mathematics and Mathematical Physics* **25**(2), 21–30.

Nemirovskii, A. and Yudin, D. B. [1983], 'Problem complexity and method efficiency in optimization'.

Nesterov, Y. [1983], 'A method of solving a convex programming problem with convergence rate $O(1/k^2)$', *Soviet Mathematics Doklady* **27**(2), 372–376.

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

Nesterov, Y. [2009], 'Primal-dual subgradient methods for convex problems', *Mathematical programming* **120**(1), 221–259.

Nesterov, Y. [2012], 'Efficiency of coordinate descent methods on huge-scale optimization problems', *SIAM Journal on Optimization* **22**(2), 341–362.

Nesterov, Y. [2013*a*], 'Gradient methods for minimizing composite functions', *Mathematical Programming* **140**(1), 125–161.

Nesterov, Y. [2013*b*], *Introductory lectures on convex optimization: A basic course*, Vol. 87, Springer Science & Business Media.

Nesterov, Y. [2015], 'Universal gradient methods for convex optimization problems', *Mathematical Programming* **152**(1-2), 381–404.

Nesterov, Y. and Shikhman, V. [2015], 'Quasi-monotone subgradient methods for nonsmooth convex minimization', *Journal of Optimization Theory and Applications* **165**(3), 917–940.

Nocedal, J. and Wright, S. J. [1999], *Numerical Optimization*, Springer.

Nova, D. and Estévez, P. A. [2014], 'A review of learning vector quantization classifiers', *Neural Computing and Applications* **25**(3-4), 511–524.

Obozinski, G. and Bach, F. [2012], 'Convex relaxation for combinatorial penalties', *arXiv preprint arXiv:1205.1240* .

Obozinski, G., Jacob, L. and Vert, J.-P. [2011], 'Group lasso with overlaps: the latent group lasso approach', *arXiv preprint arXiv:1110.0413* .

O'Donoghue, B. and Candes, E. [2015], 'Adaptive restart for accelerated gradient schemes', *Foundations of computational mathematics* **15**(3), 715–732.

Ordóñez, F. and Freund, R. M. [2003], 'Computational experience and the explanatory value of condition measures for linear optimization', *SIAM Journal on Optimization* **14**(2), 307–333.

Oymak, S. and Hassibi, B. [2010], 'New null space results and recovery thresholds for matrix rank minimization', *arXiv preprint arXiv:1011.6326* .

O'donoghue, B. and Candes, E. [2015], 'Adaptive restart for accelerated gradient schemes', *Foundations of computational mathematics* **15**(3), 715–732.

Pajor, A. and Tomczak-Jaegermann, N. [1986], 'Subspaces of small codimension of finite-dimensional banach spaces', *Proceedings of the American Mathematical Society* **97**(4), 637–642.

Pan, S. J. and Yang, Q. [2010], 'A survey on transfer learning', *IEEE Transactions on knowledge and data engineering* **22**(10), 1345–1359.

Pang, B. and Lee, L. [2005], Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales, *in* 'Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics', Association for Computational Linguistics, pp. 115–124.

Pang, J.-S. [1997], 'Error bounds in mathematical programming', *Mathematical Programming* **79**(1-3), 299–332.

Peña, J. [2000], 'Understanding the geometry of infeasible perturbations of a conic linear system', *SIAM Journal on Optimization* **10**(2), 534–550.

Peng, H., Long, F. and Ding, C. [2005], 'Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy', *IEEE Transactions on pattern analysis and machine intelligence* **27**(8), 1226–1238.

Petry, S., Flexeder, C. and Tutz, G. [2011], 'Pairwise fused lasso'.

Pfeffermann, D. and Nathan, G. [1981], 'Regression analysis of data from a cluster sample', *Journal of the American Statistical Association* **76**(375), 681–689.

Poljak, B. [1987], *Introduction to optimization*, Optimization Software.

Polyak, B. [1979], Sharp minima institute of control sciences lecture notes, moscow, ussr, 1979, *in* 'IIASA workshop on generalized Lagrangians and their applications, IIASA, Laxenburg, Austria'.

Polyak, B. T. [1963], 'Gradient methods for minimizing functionals', *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki* **3**(4), 643–653.

Polyak, B. T. [1964], 'Some methods of speeding up the convergence of iteration methods', *USSR Computational Mathematics and Mathematical Physics* **4**(5), 1–17.

Press, W. H. [1992], *The art of scientific computing*, Cambridge university press.

Rao, N., Recht, B. and Nowak, R. [2012], 'Signal recovery in unions of subspaces with applications to compressive imaging', *arXiv preprint arXiv:1209.3079* .

Raskutti, G., Wainwright, M. J. and Yu, B. [2010], 'Restricted eigenvalue properties for correlated gaussian designs', *The Journal of Machine Learning Research* **11**, 2241–2259.

Rauhut, H., Schneider, R. and Stojanac, Ž. [2017], 'Low rank tensor recovery via iterative hard thresholding', *Linear Algebra and its Applications* **523**, 220–262.

Recht, B., Xu, W. and Hassibi, B. [2008], Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization, *in* 'Decision and Control, 2008. CDC 2008. 47th IEEE Conference on', IEEE, pp. 3065–3070.

Renegar, J. [1995*a*], 'Incorporating condition measures into the complexity theory of linear programming', *SIAM Journal on Optimization* **5**(3), 506–524.

Renegar, J. [1995*b*], 'Linear programming, complexity theory and elementary functional analysis', *Mathematical Programming* **70**(1-3), 279–351.

Renegar, J. [2001], *A mathematical view of interior-point methods in convex optimization*, Vol. 3, Siam.

Renegar, J. [2014], 'Efficient first-order methods for linear programming and semidefinite programming', *arXiv preprint arXiv:1409.5832* .

Robinson, S. M. [1975], 'An application of error bounds for convex programming in a linear space', *SIAM Journal on Control* **13**(2), 271–273.

Rockafellar, R. T. [1976], 'Monotone operators and the proximal point algorithm', *SIAM journal on control and optimization* **14**(5), 877–898.

Rockafellar, R. T. [2015], *Convex analysis*, Princeton University Press.

Roulet, V., Boumal, N. and d'Aspremont, A. [2015], 'Renegar's condition number, shaprness and compressed sensing performance', *arXiv preprint arXiv:1506.03295* .

Roulet, V. and d'Aspremont, A. [2017], 'Sharpness, restart and acceleration', *arXiv preprint arXiv:1702.03828* .

Schölkopf, B., Smola, A. and Müller, K.-R. [1998], 'Nonlinear component analysis as a kernel eigenvalue problem', *Neural computation* **10**(5), 1299–1319.

Scieur, D., d'Aspremont, A. and Bach, F. [2016], Regularized nonlinear acceleration, *in* 'Advances in Neural Information Processing Systems', pp. 712–720.

Scieur, D., Roulet, V., Bach, F. and d'Aspremont, A. [2017], 'Integration methods and accelerated optimization algorithms', *arXiv preprint arXiv:1702.06751* .

Segal, M. R., Dahlquist, K. D. and Conklin, B. R. [2003], 'Regression approaches for microarray data analysis', *Journal of Computational Biology* **10**(6), 961–980.

Shanno, D. F. [1970], 'Conditioning of quasi-newton methods for function minimization', *Mathematics of computation* **24**(111), 647–656.

She, Y. et al. [2010], 'Sparse regression with exact clustering', *Electronic Journal of Statistics* **4**, 1055–1096.

Shen, X. and Huang, H.-C. [2010], 'Grouping pursuit through a regularization solution surface', *Journal of the American Statistical Association* **105**(490), 727–739.

Simon, L. [1983], 'Asymptotics for a class of non-linear evolution equations, with applications to geometric problems', *Annals of Mathematics* pp. 525–571.

Song, Q., Ni, J. and Wang, G. [2013], 'A fast clustering-based feature subset selection algorithm for high-dimensional data', *IEEE transactions on knowledge and data engineering* **25**(1), 1–14.

Steinhaus, H. [1956], 'Sur la division des corp materiels en parties', *Bull. Acad. Polon. Sci* **1**, 801–804.

Su, W., Boyd, S. and Candes, E. [2014], A differential equation for modeling nesterov's accelerated gradient method: Theory and insights, *in* 'Advances in Neural Information Processing Systems', pp. 2510–2518.

Tang, J., Alelyani, S. and Liu, H. [2014], 'Feature selection for classification: A review', *Data Classification: Algorithms and Applications* p. 37.

Tanner, J. and Wei, K. [2013], 'Normalized iterative hard thresholding for matrix completion', *SIAM Journal on Scientific Computing* **35**(5), S104–S125.

Tibshirani, R. [1996], 'Regression shrinkage and selection via the LASSO', *Journal of the Royal statistical society, series B* **58**(1), 267–288.

Topkis, D. M. [1978], 'Minimizing a submodular function on a lattice', *Operations research* **26**(2), 305–321.

Tseng, P. [2008], 'On accelerated proximal gradient methods for convex-concave optimization', *submitted to SIAM J.Optim* .

Van De Geer, S. A., Bühlmann, P. et al. [2009], 'On the conditions used to prove oracle results for the lasso', *Electronic Journal of Statistics* **3**, 1360–1392.

Vera, J. C., Rivera, J. C., Peòa, J. and Hui, Y. [2007], 'A primal–dual symmetric relaxation for homogeneous conic systems', *Journal of Complexity* **23**(2), 245–261.

Vershynin, R. [2010], 'Introduction to the non-asymptotic analysis of random matrices', *arXiv preprint arXiv:1011.3027* .

Vershynin, R. [2011], *Lectures in Geometric Functional Analysis*, In preparation.
URL: *http://www-personal.umich.edu/ romanv/papers/GFA-book/GFA-book.pdf*

Vidal, R. [2011], 'Subspace clustering', *IEEE Signal Processing Magazine* **28**(2), 52–68.

Von Luxburg, U. [2007], 'A tutorial on spectral clustering', *Statistics and computing* **17**(4), 395–416.

Vui, H. H. [2013], 'Global holderian error bound for nondegenerate polynomials', *SIAM Journal on Optimization* **23**(2), 917–933.

Wainwright, M. J., Jordan, M. I. and Duchi, J. C. [2012], Privacy aware learning, *in* 'Advances in Neural Information Processing Systems', pp. 1430–1438.

Wang, H. and Song, M. [2011], 'Ckmeans. 1d. dp: optimal k-means clustering in one dimension by dynamic programming', *The R Journal* **3**(2), 29–33.

Wibisono, A., Wilson, A. C. and Jordan, M. I. [2016], 'A variational perspective on accelerated methods in optimization', *Proceedings of the National Academy of Sciences* p. 201614734.

Wilson, A. C., Recht, B. and Jordan, M. I. [2016], 'A lyapunov analysis of momentum methods in optimization', *arXiv preprint arXiv:1611.02635* .

Yang, T. [2016], 'Adaptive accelerated gradient converging methods under holderian error bound condition', *arXiv preprint arXiv:1611.07609* .

Yen, I. E.-H., Hsieh, C.-J., Ravikumar, P. K. and Dhillon, I. S. [2014], Constant nullspace strong convexity and fast convergence of proximal methods under high-dimensional settings, *in* 'Advances in Neural Information Processing Systems', pp. 1008–1016.

Yu, L. and Liu, H. [2003], Feature selection for high-dimensional data: A fast correlation-based filter solution, *in* 'Proceedings of the 20th international conference on machine learning (ICML-03)', pp. 856–863.

Zhang, B. [2003], Regression clustering, *in* 'Data Mining, 2003. ICDM 2003. Third IEEE International Conference on', IEEE, pp. 451–458.

Zhang, H. [2017], 'The restricted strong convexity revisited: analysis of equivalence to error bound and quadratic growth', *Optimization Letters* **11**(4), 817–833.

Zhang, Y. and Yeung, D.-Y. [2010], A convex formulation for learning task relationships in multi-task learning, *in* 'Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence', AUAI Press, pp. 733–742.

Zhong, Y. and Boumal, N. [2017], 'Near-optimal bounds for phase synchronization', *arXiv preprint arXiv:1703.06605* .

Zhou, J., Chen, J. and Ye, J. [2011], Clustered multi-task learning via alternating structure optimization, *in* 'Advances in neural information processing systems', pp. 702–710.

Zhou, Z., Zhang, Q. and So, A. M.-C. [2015], l1, p-norm regularization: Error bounds and convergence rate analysis of first-order methods, *in* 'Proceedings of the 32nd International Conference on Machine Learning,(ICML)', pp. 1501–1510.

List of Figures

List of Tables

Résumé

Dans de nombreux domaines tels que l'apprentissage statistique, la recherche opérationnelle ou encore la conception de circuits, une tâche est modélisée par un jeu de paramètres que l'on cherche à optimiser pour prendre la meilleure décision possible. Mathématiquement, le problème revient à minimiser une fonction de l'objectif recherché par des algorithmes itératifs. Le développement de ces derniers dépend alors de la géométrie de la fonction ou de la structure du problème.

Dans une première partie, cette thèse étudie comment l'acuité d'une fonction autour de ses minima peut être exploitée par le redémarrage d'algorithmes classiques. Les schémas optimaux sont présentés pour des problèmes convexes généraux. Ils nécessitent cependant une description complète de la fonction, ce qui est rarement disponible. Des stratégies adaptatives sont donc développées et prouvées être quasi-optimales. Une analyse spécifique est ensuite conduite pour les problèmes parcimonieux qui cherchent des représentations compressées des variables du problème. Leur géométrie conique sous-jacente, qui décrit l'acuité de la fonction de l'objectif, se révèle contrôler à la fois la performance statistique du problème et l'efficacité des procédures d'optimisation par une seule quantité.

Une seconde partie est dédiée aux problèmes d'apprentissage statistique. Ceux-ci effectuent une analyse prédictive de données à l'aide d'un large nombre d'exemples. Une approche générique est présentée pour à la fois résoudre le problème de prédiction et le simplifier en groupant soit les variables, les exemples ou les tâches. Des méthodes algorithmiques systématiques sont développées en analysant la géométrie induite par une partition des données. Une analyse théorique est finalement conduite lorsque les variables sont groupées par analogie avec les méthodes parcimonieuses.

Mots Clés

Optimisation convexe, Borne d'erreur, Parcimonie, Acuité, Modèles structurés par partitions des données.

Abstract

In numerous fields such as machine learning, operational research or circuit design, a task is modeled by a set of parameters to be optimized in order to take the best possible decision. Formally, the problem amounts to minimize a function describing the desired objective with iterative algorithms. The development of these latter depends then on the characterization of the geometry of the function or the structure of the problem.

In a first part, this thesis studies how sharpness of a function around its minimizers can be exploited by restarting classical algorithms. Optimal schemes are presented for general convex problems. They require however a complete description of the function that is rarely available. Adaptive strategies are therefore developed and shown to achieve nearly optimal rates. A specific analysis is then carried out for sparse problems that seek for compressed representation of the variables of the problem. Their underlying conic geometry, that describes sharpness of the objective, is shown to control both the statistical performance of the problem and the efficiency of dedicated optimization methods by a single quantity.

A second part is dedicated to machine learning problems. These perform predictive analysis of data from large set of examples. A generic framework is presented to both solve the prediction problem and simplify it by grouping either features, samples or tasks. Systematic algorithmic approaches are developed by analyzing the geometry induced by partitions of the data. A theoretical analysis is then carried out for grouping features by analogy to sparse methods.

Keywords

Convex optimization, Error bound, Sparsity, Sharpness, Structured models with partitions of the data.