



HAL
open science

Active Set Algorithms for the LASSO

Manuel Loth

► **To cite this version:**

Manuel Loth. Active Set Algorithms for the LASSO. Machine Learning [cs.LG]. Université des Sciences et Technologie de Lille - Lille I, 2011. English. NNT: . tel-00845441

HAL Id: tel-00845441

<https://theses.hal.science/tel-00845441v1>

Submitted on 17 Jul 2013

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

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

T H È S E
présentée en vue d'obtenir le grade de
Docteur, spécialité Informatique
par
Manuel LOTH

**Algorithmes d'Ensemble Actif
pour le LASSO**

préparée dans l'équipe-projet Sequel commune



rédigée en anglais,

soutenue en français le 8 juillet 2011 devant le jury composé de

Président :

Marc SEBBAN, professeur des universités - Université Jean Monnet de St-Étienne

Rapporteurs :

Stéphane CANU, professeur des universités - INSA de Rouen

Patrick GALLINARI, professeur des universités - UPMC Sorbonne Universités

Examineur :

Antoine CORNUÉJOLS, professeur des universités - AgroParisTech

Directeur :

Philippe PREUX, professeur des universités - Université de Lille

Résumé

Cette thèse aborde le calcul de l'opérateur LASSO (Least Absolute Shrinkage and Selection Operator), ainsi que des problématiques qui lui sont associées, dans le domaine de la régression. Cet opérateur a suscité une attention croissante depuis son introduction par Robert Tibshirani en 1996, par sa capacité à produire ou identifier des modèles linéaires parcimonieux à partir d'observations bruitées, la parcimonie signifiant que seules quelques unes parmi de nombreuses variables explicatives apparaissent dans le modèle proposé. Cette sélection est produite par l'ajout à la méthode des moindres-carrés d'une contrainte ou pénalisation sur la somme des valeurs absolues des coefficients linéaires, également appelée norme l_1 du vecteur de coefficients.

Après un rappel des motivations, principes et problématiques de la régression, des estimateurs linéaires, de la méthode des moindres-carrés, de la sélection de modèle et de la régularisation, les deux formulations équivalentes du LASSO – contrainte ou régularisée – sont présentées; elles définissent toutes deux un problème de calcul non trivial pour associer un estimateur à un ensemble d'observations et un paramètre de sélection. Un bref historique des algorithmes résolvant ce problème est dressé, et les deux approches permettant de gérer la non-différentiabilité de la norme l_1 sont présentées, ainsi que l'équivalence de ces problèmes avec un programme quadratique.

La seconde partie se concentre sur l'aspect pratique des algorithmes de résolution du LASSO. L'un d'eux, proposé par Michael Osborne en 2000, est reformulé. Cette reformulation consiste à donner une définition et explication générales de la méthode d'ensemble actif, qui généralise l'algorithme du simplexe à la programmation convexe, puis à la spécifier progressivement pour la programmation LASSO, et à adresser les questions d'optimisation des calculs algébriques. Bien que décrivant essentiellement le même algorithme que celui de Michael Osborne, la présentation qui en est faite ici a l'ambition d'en exposer clairement les mécanismes, et utilise des variables différentes. Outre le fait d'aider à mieux comprendre cet algorithme visiblement sous-estimé, l'angle par lequel il est présenté éclaire le fait – nouveau – que la même méthode s'applique naturellement à la formulation régularisée du LASSO, et non uniquement à la formulation contrainte. La populaire méthode par homotopie (ou LAR-LASSO, ou LARS) est ensuite présentée comme une dérivation de la méthode d'ensemble actif, amenant une formulation alternative et quelque peu simplifiée de cet algorithme qui fournit les solutions du LASSO pour chaque valeur de son paramètre. Il est montré que, contrairement aux résultats d'une étude récente de Jerome H. Friedman, des implémentations de ces algorithmes suivant ces reformulations sont plus efficaces en terme de temps de calcul qu'une méthode de descente par coordonnées.

La troisième partie étudie dans quelles mesures ces trois algorithmes (ensemble actif, homotopie, et descente par coordonnées) peuvent gérer certains cas

particuliers, et peuvent être appliqués à des extensions du LASSO ou d'autres problèmes similaires. Les cas particuliers incluent les dégénérescences, comme la présence de variables linéairement dépendantes, ou la sélection/désélection simultanée de variables. Cette dernière problématique, qui était délaissée dans les travaux précédents, est ici expliquée plus largement et une solution simple et efficace y est apportée. Une autre cas particulier est la sélection LASSO à partir d'un nombre très large, voire infini de variables, cas pour lequel la méthode d'ensemble actif présente un avantage majeur. Une des extensions du LASSO est sa transposition dans un cadre d'apprentissage en ligne, où il est désirable ou nécessaire de résoudre le problème sur un ensemble d'observations qui évolue dans le temps. A nouveau, la flexibilité limitée de la méthode par homotopie la disqualifie au profit des deux autres. Une autre extension est l'utilisation de la pénalisation l_1 sur d'autres fonction coûts que la norme l_2 du résidu, ou en association avec d'autres pénalisations, et il est rappelé ou établi dans quelles mesures et de quelle façon chaque algorithme peut être transposé à ces problèmes.

Active Set Algorithms for the LASSO

Abstract

This thesis disserts on the computation of the Least Absolute Shrinkage and Selection Operator (LASSO) and derivate problems, in regression analysis. This operator has drawn increasing attention since its introduction by Robert Tibshirani in 1996, for its ability to provide or recover sparse linear models from noisy observations, sparsity meaning that only a few of possibly many explaining variables are selected to appear in the model. The selection is a result of adding to the least-squares method a constraint or minimization on the sum of absolute values of the linear coefficients, otherwise called the l_1 norm of the coefficient vector.

After recounting the motivations, principles and problematics of regression analysis, linear estimators, least-squares minimization, model selection, and regularization, the two equivalent formulations of the LASSO – constrained or regularized – are presented, that both define a non-trivial computation problem to associate an estimator to a set of observations and a selection parameter. A brief history of algorithms for solving these problems is given, as well as the two possible approaches for handling the non differentiability of the l_1 norm, and the equivalence to a quadratic program is explained.

The second part focuses on practical algorithms for solving the LASSO. An algorithm proposed in 2000 by Michael Osborne is reformulated. This reformulation consists in giving a general definition and explanation of the active set method, that generalizes the simplex algorithm to convex programming, then specifying it to the LASSO program, and separately addressing linear algebra optimizations. Although it describes the same algorithm in essence, the presentation given here aims at exhibiting clearly its mechanisms, and uses different variables. In addition to helping understand and use this algorithm that seemed to be underrated, the alternative view taken here brings light on the possibility and advantages, not foreseen by the authors, to use the method for the regularized (and more practical) problem, as well as for the constrained one. The popular homotopy (or LAR-LASSO) method is then derived from this active set method, yielding also an alternative and somewhat simplified view of this algorithm that can compute the operator for all values of its parameter (LASSO path). Practical implementations following these formulations are shown to be the most efficient methods of LASSO-path computation, contrasting with a recent study of Jerome H. Friedman suggesting that a coordinate descent method improves by far the state-of-the-art results of homotopy, in terms of speed.

The third part examines how these three algorithms (active set, homotopy, and coordinate descent) can handle some limit cases, and can be applied to extended problems. The limit cases include degeneracies, like duplicated or linearly dependent variables, or simultaneous selections/deselections of variables. The latter issue, that was dismissed in previous works, is explained and given a

simple solution. Another limit case is the use of a very large, possibly infinite number of variables to select from, where the active set method presents a major advantage over the homotopy. A first extension to the LASSO is its transposition in online learning settings, where it is necessary or desirable to solve for a growing or changing observation set. Again, the lack of flexibility of the homotopy method discards it in profit of the other two. The second extension is the use of l1 penalization with other loss function than the squared residual, or together with other penalization terms, and we summarize or state to which extent and how each algorithm can be transposed for these problems.

Rapports

Rapport de Stéphane Canu,
Professeur à l'institut national des sciences appliquées de Rouen,
sur les travaux présentés par Monsieur Manuel Loth
pour lui permettre d'obtenir le grade de docteur de l'Université de Lille
Nord de France, travaux intitulés :

Algorithmes d'ensemble actif pour le LASSO

Le problème abordé par Monsieur Manuel Loth dans sa thèse est celui de la résolution efficace du problème de LASSO. L'angle d'attaque choisi est celui de l'algorithme des contraintes actives qui permet d'utiliser judicieusement la nature très parcimonieuse de la solution. La principale contribution de ce travail est ECON, un algorithme et le logiciel le mettant en oeuvre pour résoudre le problème du LASSO. Cet algorithme permet notamment le traitement en ligne ce qui permet de l'utiliser dans le cadre de l'apprentissage par renforcement, ce qui est une approche nouvelle, efficace et prometteuse.

Le manuscrit présenté est constitué de trois chapitres encadrés par une introduction et une conclusion.

Le premier chapitre présente le problème d'estimation du LASSO dans le cadre général de l'apprentissage et de la régularisation. Ce chapitre est intéressant, agréable à lire et assez complet. Cependant le lasso aurait pu être présenté sous la forme standard d'un programme quadratique ce qui aurait permis une présentation unifiée, plus synthétique et mieux en relation avec l'état de l'art très important sur le sujet dans le domaine de l'optimisation. Quelques travaux de références auraient pu être cités comme : Mark Schmidt, Glenn Fung, and Romer Rosales. *Fast optimization methods for l_1 regularization: A comparative study and two new approaches*. In Proceedings of European Conference on Machine Learning, 2007 et G. X. Yuan, K.W. Chang, C. J. Hsieh, and C. J. Lin. *Comparison of optimization methods and software for large-scale l_1 -regularized linear classification*. *Journal of Machine Learning Research*, 2010. Enfin, dans le dernier paragraphe, l'utilisation de sous gradients, maintenant assez classique dans le domaine, permet une autre compréhension du problème (voir par exemple M. J. Wainwright. *Sharp thresholds for high-dimensional and noisy sparsity recovery using l_1 -constrained quadratic programs*. In Proc. Allerton Conference on Communication, Control and Computing, October 2006).

Le deuxième chapitre trois présente trois méthodes permettant de résoudre le programme quadratique associé au LASSO : la méthode des contraintes actives, l'homotopie (avec les détails liés de la mise à jour de la factorisation de Cholesky) et la méthode de Gauss Seidel. Les autres approches du problème : les algorithmes de type proximal, les méthodes de Newton et celle de points intérieurs (entre autres) ne sont pas traités. Cependant, les méthodes présentées montrent qu'un bonne maîtrise des différents aspects de l'utilisation de ces algorithmes et de leur mise en oeuvre. De plus, les résultats rapportés dans ce chapitre montrent l'intérêt des méthodes de type contraintes actives pour ce genre de problème. Ce type de résultat est tout à fait cohérent avec ce que l'on sait dans le domaine des SVM, on l'on retrouve un programme quadratique tout à fait analogues. Les deux domaines (SVM et LASSO) devraient sans doute mieux bénéficier de leurs avancées réciproques. Il manque enfin à ce chapitre une conclusion générale permettant de mieux exploiter les résultats obtenus.

Le troisième chapitre propose de combiner les trois approches pour éliminer leurs inconvénients respectifs dans une nouvelle approche baptisée ECON. Le premier d'entre eux est la dégénérescence, bien connue des utilisateurs de l'homotopie puisqu'elle a été documentée dès sa première publication dans les années 50 (voir les travaux de Markowitz sur le sujet). La question des très grands jeux de données est ensuite abordée. Mais la partie la plus intéressante et la plus innovante à mon sens est le traitement du cas de l'apprentissage séquentiel avec l'application à l'apprentissage par renforcement de la méthode du LASSO. Il s'agit là d'une perspective très innovante et tout à fait intéressante en cohérence avec les travaux de l'équipe dans laquelle cette thèse s'est déroulée. La section suivante traite des possibilités d'extensions à d'autres critères. On regrette que les résultats n'aient pas été mis en rapport avec les dernières avancées de l'état de l'art (voir par exemple les travaux de Karasuyama et Takeuchi, « *Suboptimal Solution Path Algorithm for Support Vector Machine* » disponibles sous Arxiv preprint arXiv:1105.0471 et acceptés à ICML 2011). Ces travaux ont été concrétisés par l'écriture d'un logiciel ECON et d'un ensemble d'expériences montrant l'intérêt de l'approche proposée.

En conclusion, le mémoire présenté est bien construit, bien illustré et contient un algorithme intéressant (ECON). Les résultats présentés dans cette thèse ont été régulièrement publiés dans des conférences du domaine. Une de ces communications publiée en 2007 est citée 32 fois d'après Google Scholar ce qui indique un intérêt de la communauté pour ces travaux. Ces résultats devraient maintenant être valorisés par une publication dans une revue internationale. En conséquence, le travail de Monsieur Manuel Loth présente à mes yeux les qualités requises pour lui permettre d'obtenir le grade de docteur de l'Université de Lille Nord de France et je suis favorable à la présentation orale de ses travaux.



Fait à Rouen, le 18 juin 2011,

Stéphane Canu



Rapport concernant le manuscrit de thèse de Manuel LOTH intitulé « algorithmes d'Ensemble Actif pour le LASSO »

Le travail de M. Loth porte sur les méthodes d'apprentissage de représentations parcimonieuses de la famille LASSO (Least Absolute Shrinkage Operator). Ces méthodes devenues populaires en quelques années, pour la sélection de caractéristiques, ont suscité de très nombreux travaux en statistique, apprentissage et traitement du signal. Dans son manuscrit, M. Loth propose de réexaminer certains des algorithmes développés pour le LASSO qui sont restés un peu dans l'ombre d'autres méthodes. Il en donne une reformulation qui lui sert de base ensuite pour proposer des adaptations et extensions de ces algorithmes.

Le chapitre 1 introduit, dans le cadre général de la régression, le LASSO, et donne une historique des travaux marquants dans le domaine, notamment la formulation initiale de Tibshirani en 96, les travaux de Osborne en 2000 qui serviront de base pour cette thèse et l'algorithme de descente par coordonnée de Friedman en 2007. Il donne ensuite une reformulation de la définition du LASSO qui étend légèrement la définition originale en considérant, un espace de caractéristiques plus général, une normalisation des paramètres ou des caractéristiques, un nombre éventuellement infini de caractéristiques.

Le chapitre 2 décrit le cœur du travail qui consiste à adapter une méthode d'ensemble actif pour le problème du LASSO. M. Loth commence par présenter une méthode générale d'optimisation par ensemble actif. Il caractérise la solution LASSO dans le cas d'un ensemble actif optimal puis dérive un algorithme d'optimisation dans le cas d'une fonction convexe. Il montre ensuite comment l'algorithme peut s'adapter au LASSO. Une seconde contribution consiste à montrer l'équivalence entre les formulations contraintes et régularisées du LASSO. Par la suite, c'est cette version régularisée qui sera principalement utilisée dans la thèse. La troisième contribution est la dérivation de l'algorithme d'Osborne dit « homotopy method » à partir d'une formulation par ensemble actif. Cette reformulation conduit à un algorithme relativement simple. La complexité des deux algorithmes est analysée et une implémentation utilisant des décompositions matricielles est présentée. Le comportement de trois méthodes (ensemble actif, homotopie et descente par coordonnée qui sert de référence) est illustré sur un exemple.

Le chapitre trois présente trois contributions principales : l'analyse de cas de dégénérescences du LASSO, le cas d'un nombre infini de caractéristiques et l'extension à l'apprentissage en ligne des trois algorithmes précédents.

Les cas de dégénérescence étudiés concernent l'existence de solutions multiples (plusieurs sous-ensembles de caractéristiques peuvent être solution du LASSO), et uniquement dans le cas de la méthode par homotopie, la violation simultanée de plusieurs contraintes.

Pour les solutions multiples, M. Loth montre que la solution LASSO est unique sauf quand il existe des dépendances affines entre caractéristiques. Pour le cas de violation simultanée, il propose de combiner la méthode par homotopie et la méthode par ensemble actif.

Une seconde contribution concerne l'expression et l'implémentation des algorithmes LASSO dans le cas d'un nombre infini de caractéristiques. M. Loth énumère les problèmes pratiques rencontrés. Il décrit brièvement sa méthode « ECON » développée pour des caractéristiques définies comme des fonctions paramétriques et donc potentiellement non dénombrables. Il expose des solutions pratiques développées dans ce cadre, comme l'approximation du chemin de régularisation ou encore une fois la combinaison méthode par homotopie - méthode d'ensemble actif. Il complète cette description par deux méthodes d'optimisation pratiques qui remplacent l'optimisation fine usuelle en apprentissage par une série d'estimations moins précises qui peuvent être obtenues par échantillonnage adaptatif des exemples ou par gradients multiples. La dernière contribution concerne l'apprentissage séquentiel. M. Loth propose une adaptation des algorithmes LASSO en exploitant une idée similaire aux moindres carrés récurrents.

Finalement, quelques expériences sont effectuées avec l'algorithme ECON sur des petits exemples de classification et un exemple de régression.

M. Loth a réalisé un travail de bonne qualité. Il a obtenu des résultats originaux, qui ont donné lieu à plusieurs publications. Sa thèse représente un travail solide. Je donne un avis favorable à la soutenance de ce travail pour l'obtention du grade de docteur de l'université de Lille 1.

Paris le 19-06-2011

P. Gallinari

Professeur



Surface address: University Paris VI, 4 Place Jussieu, 75252 Paris cedex 05, France

Tel: (33-1) 44 27 73 70 - Fax (33-1) 44 27 70 00 - E-mail: Patrick.Gallinari@lip6.fr

Rapport de soutenance de thèse

Nom et prénom : Mr LOTH MANUEL

Titre de la thèse :
ALGORITHMES D'ENSEMBLES ACTIFS POUR LE LASSO

Discipline : INFORMATIQUE

Date de soutenance : 08/07/2011

Le jury a apprécié la qualité de la soutenance de Manuel Loth durant laquelle il a présenté ses travaux de thèse situés dans le domaine de l'optimisation par algorithmes avec contraintes actives pour résoudre le Lasso. Manuel Loth a su exploiter avec pertinence des animations graphiques pour introduire des concepts d'optimisation non triviaux utilisés dans ses recherches. L'exposé a montré la maturité scientifique du candidat et sa connaissance approfondie des méthodes d'optimisation, confirmant ainsi l'impression déjà donnée par la lecture du manuscrit, clair et rédigé en langue anglaise.

Enfin, le jury a apprécié la passion avec laquelle le candidat a essayé durant sa thèse de comprendre des problèmes difficiles, et la qualité et la justesse des réponses qu'il a apportées aux différentes questions posées.

Pour toutes ces raisons, le jury a décidé à l'unanimité de décerner à Manuel Loth le grade de docteur en informatique de l'Université de Lille 1.

Mention :

Honorable

Très Honorable

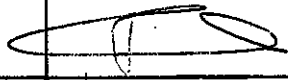



L'USTL n'attribue plus de mention "Très Honorable avec les Félicitations" (par décision du Conseil Scientifique du 15 juin 2007)


Nombre de pages du rapport : 1

Jury :

Président : D. SEBBAN

Signatures des membres

PREUX Philippe	
GALLINARI Patrick	
CANU Stéphane	
CORNUEJOLS Antoine	

SEBBAN Marc	

Remerciements

Je tiens avant tout à remercier mon directeur de thèse Philippe Preux, pour son soutien sans faille dans les moments les plus difficiles. Je suis très reconnaissant à Stéphane Canu et Patrick Gallinari de m'avoir chacun accordé une lecture attentive et un rapport pertinent de cette thèse, ainsi qu'un échange enrichissant lors de sa soutenance; je remercie également Antoine Cornuéjols et Marc Sebban de leur présence, remarques et encouragements lors de cette soutenance.

Enfin, je remercie l'INRIA pour son appréciable soutien financier et logistique, et la Région Nord – Pas de Calais pour son soutien financier.

Les autres remerciements ont été formulés directement aux personnes concernées 😊.

Contents

1	The LASSO	31
1.1	Regression	31
1.1.1	Estimating the conditional mean	33
1.1.2	Practical models	36
1.1.3	Linear regression	37
1.1.4	Loss functions	41
1.1.5	Regularization	44
1.2	The Least Absolute Shrinkage Operator	45
1.2.1	Algorithms for solving the LASSO	50
1.2.2	Extended definition	51
1.2.3	LASSO programming	53
2	Active set Algorithms	55
2.1	Solving the constrained problem on a given active set	55
2.1.1	Optimality of an active set	56
2.1.2	The computation step	57
2.2	The active set descent method	58
2.2.1	The active set descent method for convex programming	59
2.2.2	active set descent for the LASSO	66
2.2.3	Active set descent for the regularized LASSO	75
2.3	The homotopy method	78
2.3.1	Computing the first unfeasibility point	79
2.3.2	Computing the first insufficiency point	80
2.3.3	Jumping to the next break point	81
2.4	Complexity	83
2.4.1	Sequential least-squares	85
2.4.2	Cholesky decomposition	85
2.4.3	Cholesky rank one update	87
2.4.4	Feature addition	88
2.4.5	Feature subtraction	89
2.5	The coordinate descent method	90
2.5.1	The one-dimensional LASSO	91
2.6	Experiments	92
2.6.1	Illustration of LASSO estimators	92
2.6.2	Speed trials	95
2.6.3	LASSO path and descent paths	95

3	Beyond the Simple LASSO	103
3.1	Degeneracies	104
3.1.1	Multiple solutions	105
3.1.2	Simultaneous status changes	110
3.2	Very large feature sets	119
3.2.1	Non-countable sets	119
3.2.2	Practical optimization	137
3.3	Sequential Learning	141
3.3.1	Online Learning	143
3.3.2	Non-stationary problems	146
3.3.3	Reinforcement Learning	147
3.4	Alternative loss functions	148
3.4.1	Alternative penalties	149
3.4.2	Alternative residual losses	153
3.5	ECON Experiments	156
3.5.1	ECON with homotopy and adaptive penalization	156
3.5.2	ECON with ASD	159
	Bibliography	167

List of Figures

1.1	Artificial data, simulating the height data of (Galton, 1886) . . .	34
1.2	Different ℓ^q -norms in \mathbb{R}^2 , for $q = 1/5, 1/2, 1, 2$; ℓ^1 is the first convex one (with q increasing), and also the last one with an “angular” property of presenting vertices where one or several attributes are zero.	48
1.3	Interaction of the ℓ^2 -norm of the residual (loss) and ℓ^1 -norm of the coefficient, in the coefficient-space view.	49
2.1	Geometrical representation of the computation of a solution on a known, or assumed, optimal active set $\{+\phi_1, +\phi_2\}$	60
2.2	From an assumed optimal active set $\{+\phi_1, +\phi_2\}$, disagreement of the regularized least-square with the sign hypothesis, indicating a better solution β_- on the reduced active set $\{+\phi_1\}$	61
2.3	An example of the continuous, decreasing, and piecewise linear relation between the parameters λ and t of equivalent constrained and regularized forms of a LASSO problem.	75
2.4	Average numbers of the most costly tasks required to reach a LASSO solution, as a function of the cardinality of the solution (k), for random and unstructured problems.	86
2.5	LASSO and restricted-least-squares (least-squares on active features) estimators for different values of λ , for noisy observations of a sinusoidal function, and 10000 Gaussian features.	94
2.6	Characterization of the sequence of estimators generated by the homotopy, active set descent, and cyclical coordinate descent algorithms, on the sinusoidal problem illustrated in figure 2.5. . . .	99
2.7	Characterization of the sequence of estimators generated by the homotopy, active set descent, and cyclical coordinate descent algorithms, on the speed trials problem of section 2.6.2, with $n = 200$ and $p = 10000$	100
2.8	The bottom-right plot of figure 2.6 with logarithmic scale.	101
3.1	LASSO and restricted-least-squares estimators for different values of λ , for noisy observations of a sinusoidal function, and 10000 Gaussian features, with features penalizations inversely proportional to their bandwidth σ	129
3.2	LASSO and restricted-least-squares estimators for different values of λ , for noisy observations of a sinusoidal function, and 10000 Gaussian features, with features penalizations inversely proportional to $\log(1 + \sigma)$	130

3.3	For the sinusoidal problem of figures 2.5, 3.1 and 3.2, corresponding to different feature penalizations, plot of the squared residual between the restricted-least-squares estimator and, respectively, the noisy outputs (empirical loss) and the model (true loss), against each step of the regularization path.	131
3.4	A problem, described in section 3.2.1.5, for which the active set of the LASSO solution evolves continuously with λ	137
3.5	Regularization paths of the problem described in section 3.2.1.5 and figure 3.4, for different discretization levels, and for the whole continuous set of features.	138
3.6	Evolution of the function optimized by the homotopy (top: value of λ for which a feature reaches equi-correlation) and ASD (bottom: features correlation), on the sinusoidal function of section 2.6.1.	142
3.7	An example of the behaviour of the "next-activation" function optimized by the homotopy, in the immediate neighbourhood of an active feature (circled point).	143
3.8	Density functions of standard Normal and Laplace distributions, that are the assumed distributions of the noise for, respectively, the least-squares and least-absolute-deviations estimators.	154
3.9	Results of a light source simulation, when using interpolation or an ECON model.	160

List of Tables

2.1	Speed trial experiments with the same settings as in (Friedman et al., 2007).	96
3.1	Comparison of different regression methods in terms of parameters.	126
3.2	Comparative results of ECON on classification problems	158
3.3	Comparative results of ECON on regression problems.	159
3.4	mean minimum prediction errors (percentage), and standard deviations, when using ECON with Gaussian-DIAG features and ASD algorithm. The results are consistently better than when using the homotopy algorithm, as in section 3.5.1.2 (table 3.2)	161
3.5	mean minimum MSE over 100 trials, and standard deviations, on the three Friedman functions, using ASD with different features dictionaries.	163

List of Algorithms

1	active set descent method for convex programming	63
2	active set descent for differentiable canonical convex programming	65
3	NNLS algorithm as given in (Lawson and Hanson, 1974)	66
4	active set descent for the linear LASSO (predictors $\in \mathbb{R}^p$)	70
5	active set descent for the featurized constrained LASSO (arbitrary feature set)	72
6	active set descent for the regularized LASSO	77
7	Homotopy method for the regularized LASSO	82
8	Cyclic Coordinate Descent for the regularized LASSO	92
9	Non-Negative Forward Stepwise	98
10	Cycle-safe homotopy method for the regularized LASSO	118
11	Active Set Coordinate Descent for the regularized LASSO	121
12	ECON: approximate homotopy method for the regularized LASSO, with adaptive penalization	134
13	Mixed algorithm: approximate homotopy method for the regularized LASSO, with corrections by active set descent.	135

Introduction

The domain of this thesis is *Machine Learning*, and more specifically *supervised learning*. Machine learning (see (Alpaydin, 2005) or (Cornuéjols and Miclet, 2010)(french) for recent overviews) emerged as a field in the 1980's following the first workshop on the subject. According to one of its founders Tom M. Mitchell (Mitchell, 2006), the field seeks to answer the question

How can we build computer systems that automatically improve with experience, and what are the fundamental laws that govern all learning processes?

Supervised learning is the most common and exemplary setting, in which experience consists in a set or sequence of input/output pairs, and the task is to predict a subsequent output given the sole input. Performance is then assessed by some measure of difference between the predicted and real outputs. Many topics of interest for machine learning, especially in this supervised setting, have a lot in common with problems studied long before the advent of programmable computing machines, in the field of statistics. The focus in statistics was, and still is, not so much on prediction as on modelization of the relation between the two variables. The term *Statistical Learning* captures well the intersection of both approaches, that is, practical and algorithmic aspects of statistics, or statistically-grounded methods of learning. A review can be found in the book "The Elements of Statistical Learning" (Hastie et al., 2008).

The learning task is referred to as *regression* when the output is continuous (belongs to a continuous set, typically \mathbb{R}), and the model, or prediction function, can be estimated by a continuous function of the inputs. Statistical laws, as well as computational considerations, motivate the search of a model among linear combinations of the input's features. These features can be the original, natural attributes of the object of study, for example the heights of both parents for predicting the height of their offspring, when it is reasonable to assume a linear relationship. They can also be artificial features, that typically quantify a notion of proximity of the input from a fixed point: the inputs are re-described in terms of how much they are close to a list of reference points. The number of such features can be large and, combined with an appropriate measure of closeness, this permits a simple linear combination of these features to correspond to a more complicated function of the original features.

Whether natural or artificial, the features may be unnecessarily numerous. A natural feature can have no correlation at all with the output, and perturb the evaluation of a prediction function. When evaluating a nonlinear relationship by the means of artificial features, they should be numerous to enlarge the choice of models, but this greater modelling power can bring computational issues, and also encourages traditional methods to fit the observations too closely (overfitting);

this is not desirable because these observations are usually corrupted by noise, and the prediction should be able to generalize from a few observations rather than simply reproducing them. Dedicated extensions to a method in order to ensure this generalization property is referred to as *regularization*.

Associated to this most common class of linear models is the simple, practical, and well grounded method of *least-squares*, that chooses as a model the one minimizing the *squared residual*, that is the sum of squares of the differences between the observed outputs and the corresponding predictions of the model. The least-squares model can be identified by a straightforward computation, in which a square matrix of size the number of features must be inverted.

A regularized extension to least-squares was proposed in (Tibshirani, 1997) as the Least Absolute Shrinkage and Selection Operator (LASSO), that minimizes the squared residual under an additional constraint on the amplitude of the linear coefficients of the model: the sum of their absolute values (ℓ^1 - norm of the coefficient vector) must be lower than a given threshold. This constraint has a selection effect: it limits the number of nonzero coefficients. This selection property is profitable by itself, for simply identifying the relevant variables, or easing computations, and preventing overfitting. Also, this specific constraint presents two advantages over the somewhat more natural constraint of limiting explicitly the number of non zero features (ℓ^0 -norm): it is better in terms of generalization, and its computation is easier, the ℓ^0 regularization yielding a combinatorial complexity.

However, computing a LASSO model is not as straightforward as for the simple least-squares model. Indeed, it mixes continuous and discrete concepts: minimization of the residual over the linear coefficients, and selection of features. Therefore, different algorithms have been proposed for solving the LASSO problem. An elegant publication (Efron et al., 2004) has popularized the LAR-LASSO algorithm, that is able to compute the solutions for all possible values of the constraint; the LASSO itself, as well as the use of the same constraint over different operators than least-squares, has since become widely popular in both applications and research. Research, however, did not focus much on the algorithmical aspect of the problem, mostly taking as a base the LAR-LASSO algorithm as it was formulated. Noticeable exceptions are the propositions of dedicated gradient-based methods (Kim et al., 2007), and coordinate-descent algorithms (Friedman et al., 2010). The first kind does not possess the appealing simplicity of the LAR-LASSO, that facilitates its understanding, implementation, analysis, or robustness. However, they seem to perform better in large scale problems, given a proper, optimized implementation. The coordinate-descent approach presents both aspects: simplicity and scalable efficiency. It was initially proposed in (Fu, 1998), soon after the LASSO itself, but the interest for it grew only recently, after a generalization and nice presentation in (Friedman et al., 2010). The same phenomenon actually happened for the LAR-LASSO, that was a larger-perspective presentation of the *homotopy* algorithm previously presented in (Osborne et al., 2000a).

One aspect of this thesis is to provide a somewhat similar renewal presentation of a second algorithm proposed in (Osborne et al., 2000a), denoted as “an active set method”. Several misunderstandings surround this algorithm: it is the same idea that was behind one of the algorithms proposed in Tibshirani’s original paper, but the adaptation was different and suboptimal. In most reviews of LASSO algorithms, the active set method is not mentioned, or wrongfully explained or analyzed. The authors themselves appear not to have fully foreseen the settings to which the method can be applied. Yet the algorithm is simple, efficient, and appears to be more flexible and robust than the homotopy / LAR-LASSO algorithm. Therefore, we try to present it here in what seems to us to be the simplest formulation. We will try to rely on as few existing frameworks and theories as possible, to provide a standalone understanding of both the problems and algorithms. We also re-explain the homotopy algorithm in the same fashion, as well as the coordinate descent method.

Such simplified presentations do not simply help the understanding and implementation, possibly to a broader audience. It permits to give a clearer view of the problematics, that is fruitful in several ways. The relation between the homotopy and the active set method appears more distinctly and allows their combination for handling special cases that were previously ignored or coarsely tackled. Far from being exotic situations, these cases happen in practise, and their proper handling is necessary to provide “black-boxed” implementations of the algorithms.

The derivation of the active set method for an alternative formulation of the LASSO appears naturally as a simpler algorithm, whereas it used to be thought to be more complex. The variables involved when translating the conceptual method to specific algorithms give simpler and more efficient implementations.

More generally, the simplified point of view that we try to take on the subject allows us to by-pass existing and popular frameworks like kernel methods: rather than considering such existing methods and juxtaposing questions of kernel choice, feature selection, multiple kernels, we propose to simply apply the LASSO to an infinite set of features, that may also correspond to an infinite number of kernel functions, without relying on these infinities to be reduced to finiteness by kernel properties. Enlightened intuition and experimental evidences agree on the fact that approximation in the face of infinity can give, in the present case, better results than finite reductions.

This thesis is thus focused on the sole LASSO, (not on similar regression operators), presented together with its context and motivations in chapter one. It concentrates on the algorithmic aspects of its computation, through the presentation in chapter two of the three algorithms that qualify as *active set algorithms*: they solve successively the LASSO on a reduced set of variables, the others being set as constant. Special cases and extensions of these algorithms and the LASSO itself are the object of the third and last chapter.

General notations

Throughout this thesis, the following notation conventions will be used:

- a vector in \mathbb{R}^n is denoted by a lowercase bold letter and its components by the same regular-sized letter with a subscript, e.g.

$$\mathbf{v} = \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix};$$

- a bold uppercase letter denotes a matrix: \mathbf{M} ;
- the transpose of a matrix or vector is denoted by \mathbf{v}^\top ;
- for the set of functions from a set A to a set B , we mostly use the arrow notation, and occasionally the exponential one:

$$f : A \rightarrow B \quad \iff \quad f \in B^A;$$

- given a set A , 2^A denotes the set of its subsets:

$$2^A = \{B \mid B \subset A\};$$

- $\text{minimize}_{(a,b) \in 2^{A \times B}} f(a,b)$, slightly abusively, means that a minimization is performed over all possible subsets a of A , each member of which is associated to all possible values b of B , the abusive part being that, strictly speaking, a subset of $A \times B$ can contain identical members of A , associated to different values of B . For example, if $A = \{a_1, a_2\}, B = \{1, 2\}$, we consider

$$\begin{aligned} & \emptyset, \\ & \{(a_1, 1)\}, \{(a_1, 2)\}, \{(a_2, 1)\}, \{(a_2, 2)\} \\ & \{(a_1, 1), (a_2, 1)\}, \{(a_1, 1), (a_2, 2)\}, \{(a_1, 2), (a_2, 1)\}, \{(a_1, 2), (a_2, 2)\} \end{aligned}$$

but not $\{(a_1, 2), (a_1, 1)\}$, and note them as a couple (a, b) where a is a subset, and b is an associated vector:

$$(a = \{a_1, a_2\}, b = (2, 2))$$

- $\max_{x \in \mathcal{X}} (f(x))_{<a} = \max_{\substack{x \in \mathcal{X} \\ f(x) < a}} f(x)$ maximizes f while ignoring value larger than a , and similar notations are used for minimizations and other conditions; when no value of x meets the condition, the result is considered to be *null*; by convention, a *null* value is ignored in all comparisons, e.g. $\max(\text{null}, 2) = 2$, and it can be verified that no *null* value can be propagated up to being involved in an arithmetic operation, in the algorithms presented in this thesis;

- $(y^*, x^*) \in \min, \arg \min_x f(x) \iff \begin{cases} x^* & \in \arg \min_x f(x) \\ y^* & = f(x^*) \end{cases}$
- where it may not cause confusion, $\mathcal{S} \cup \{x\}$ can be simply noted as $\mathcal{S} + x$.

The LASSO

The Least Absolute Shrinkage Operator (LASSO) was proposed in (Tibshirani, 1996) as a technique for linear regression. Linear regression is itself a specific technique of regression, and this thesis focuses on techniques for computing this operator. This introductory chapter precises this hierarchy of problems with their settings, motivations and notations. Particular attention is given to the LASSO itself and algorithms for solving it, techniques for handling the ℓ^1 -norm, and a generalized definition of the LASSO.

1.1 Regression

Regression [analysis] is generally defined as the task of studying the relationship between a real dependent variable and one or more independent variables.

The reference to dependency is mostly a way of defining the task as modelling the former as a function of the others, that is study the distribution of one variable conditioned on the others, and does not necessarily imply statistical assumptions. The independent variables are commonly referred to as the *predictors*, and the dependent one as the *response*.

The reference to multiple predictors can be thought as an inheritance of this domain's history and relationship to experiment. Typically, p different measurements are conducted on n objects of a group or one object in n different states, and a regression task consists in modelling the dependency of one of the p variables on one or more of the others. However, once a problem has been chosen, one might as well see the predictors as one variable.

A general and formal definition of the regression problem can be:

Definition 1. *Let $Z = (X, Y)$ be a random variable in some product set $\mathcal{X} \times \mathcal{Y}$, with $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$. Regression analysis consists in studying the conditional distribution $P(Y|X)$ based on a finite number of observations $(x_i, y_i)_{i=1, \dots, n}$ and prior assumptions.*

Throughout this chapter, we will be using as an illustrating example an analytically defined variable of the same nature as the object of study in (Galton, 1886). This article was the first to study the phenomenon of *regression towards the mean*, that consists in the tendency of off-springs to exhibit a lower deviation from the mean than their genitors on a given quantifiable property. The term *regression* thus originally referred to the general form of the relationship between these two variables, and the work in (Galton, 1886) consisted in its closer and

numerical analysis. The expression *regression analysis* was later applied to more general studies of the same nature, independently of any notion of regression in the results. In the *Machine Learning* community, the sole word regression is generally used and refers to the cases where $\mathcal{X} = \mathbb{R}$, whereas the term *classification* is used when \mathcal{X} is finite, which brings specific problematics and solutions.

The precise case analyzed in (Galton, 1886) was the relationship between the physical height of 930 adult human subjects (Y) and the average height of their two parents (X), after a preliminary scaling for the feminine subjects. The marginal distributions $P(X)$ and $P(Y)$ are similar in both shape and mean, and, as one can expect, more or less Gaussian, indicating – not surprisingly – that, in the absence of extraneous evolutions, the distribution of human height is stable throughout the generations. Yet the results suggest that although X and Y are approximately jointly Gaussian, their correlation coefficient is strictly and significantly lower than 1. Such a variable $Z = (X, Y)$ would be characterized by a bivariate Gaussian distribution

$$P(Z = \mathbf{z}) \propto \exp\left(-\frac{1}{2}(\mathbf{z} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{z} - \boldsymbol{\mu})\right), \quad (1.1)$$

where

$$\mathbf{z} = \begin{pmatrix} x \\ y \end{pmatrix}, \boldsymbol{\mu} = \begin{pmatrix} \mu \\ \mu \end{pmatrix}, \boldsymbol{\Sigma} = \begin{pmatrix} \sigma^2 & \rho\sigma^2 \\ \rho\sigma^2 & \sigma^2 \end{pmatrix},$$

μ being the theoretical mean height of a human being, σ the theoretical standard deviation from this mean, and $\rho < 1$ characterizing the regression phenomenon (in the original meaning of the term). In scalar notations, eq. (1.1) can be written

$$P(X = x, Y = y) \propto \exp\left(-\frac{1}{2} \frac{\left(\frac{x-\mu}{\sigma}\right)^2 + \left(\frac{y-\mu}{\sigma}\right)^2 - 2\rho \frac{(x-\mu)(y-\mu)}{\sigma^2}}{1 - \rho^2}\right) \quad (1.2)$$

Figure 1.1 shows artificial data obtained by drawing samples from such a distribution. This reproduces observations similar to that of the ones analyzed by Galton, by assuming that such a perfectly defined simple model is underlying the heights of human beings. Naturally, the height of a human being is not *generated* by an abstract random process based on a continuous parametric distribution, and the very concepts of random, distribution, or continuity do not have a physical reality. But nor are they arbitrary simplifications that empirically fit the observations of natural phenomena, but rather are emerging at macroscopic levels from simple interactions at microscopic levels, in a consistent way, and following coherent theories. Without disserting much further on that point, let us give an example of a fundamental law that enlightens the relationship between natural observations and theoretical concepts and models:

Let us consider n multiple independent binary events, represented by a number X_i being -1 or $+1$, for $i = 1, \dots, n$, and their aggregation at a higher level by summing these numbers, and scaling by n , obtaining a number $Y = \frac{1}{n} \sum_{i=1}^n nX_i$.

This mathematical setting, although it does not model something in particular, can be related, from its simplicity, to some physical reality, for example at a quantic level. The binary alternative for X_i 's hardly qualifies as random, or can be considered as primal randomness: the process is simply not observable closer than noting the frequencies of its results. To the 2^n possible situations correspond $2n$ possible rational values for Y , occurring increasingly frequently as they are closer to zero. The view of the mind of taking n to infinity produces the switches from \mathbb{Q} to \mathbb{R} , finiteness to continuity, case enumerations and frequencies to probabilities and distributions, all of which are theoretical but yield useful tools and theories of simpler form to manipulate quantities that are originally discrete. The theoretical continuous distribution of Y obtained in the limit is the standard normal distribution.

Once the probability notions have been established, the convergence to a normal distribution can be proven to be a general law by the *central limit theorem*, stating that the mean of a set of variables having finite mean and variance and verifying one of several weaker conditions tends to be normally distributed as the number of variables increases. This law is self-reproducing, self-reinforcing: the mean of many approximately normal variables tends even more to normality. This explains the ubiquity of this distribution for variables observed in nature or economics, and therefore an informal version of this theorem, stating that when a variable is determined by many factors, it is well modelled by a normal distribution, is often used, as we did for artificially reproducing the height data.

So the quasi-equivalence between generating data from a relatively simple artificial model, and the actual generative process beyond real-world observations, is well grounded, from this property that simple organization often emerges at macroscopic levels from numerous interactions at lower levels.

The task of an experimenter is to identify such a model or some of its characteristics, from the sole observations, and also some assumptions and restrictions on the type of possible models.

1.1.1 Estimating the conditional mean

The response Y is generally a real number, and the study of $P(Y|X)$ often concentrates on its expected value $E(Y|X)$, which is a function $f : \mathcal{X} \rightarrow \mathbb{R}$. This study mostly consists in providing an estimation of this function given the observations, as close as possible to the real value and if possible converging to it as the number of observations goes to infinity. Such an *estimator* is part of a general model describing (X, Y) . The problem of regression is often stated, especially in machine learning, as approximating a function f from noisy samples. Two formulations of this task can be made:

Definition 2. *Let X, Y be joint random variables in $\mathcal{X} \times \mathbb{R}$, and $(x_i, y_i)_{i=1, \dots, n}$ n independent observations.*

Let $f(x) = E(Y|X = x)$. Find an estimator $\hat{f} \in \mathbb{R}^{\mathcal{X}}$ close to f .

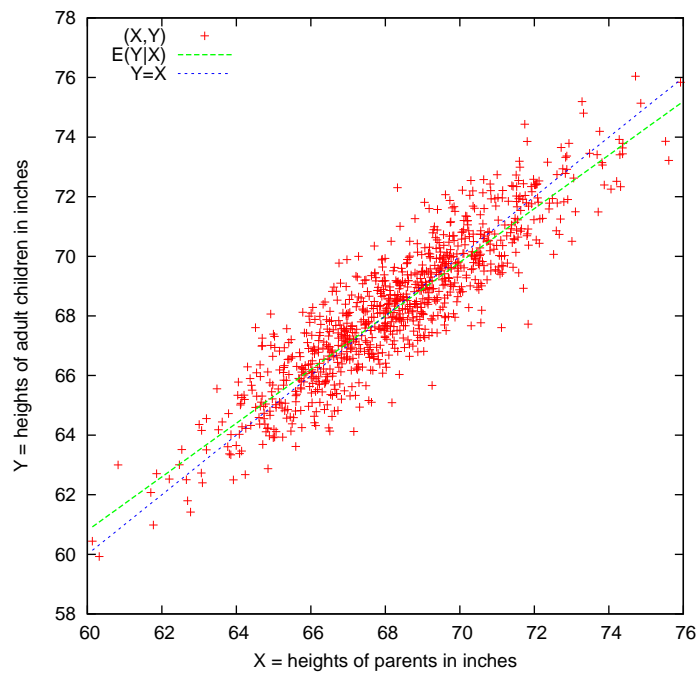


Figure 1.1: Artificial data, simulating the height data of (Galton, 1886)

Definition 3. Let $(x_i, y_i)_{i=1, \dots, n}$ be n noisy observations of a function $f : \mathcal{X} \rightarrow \mathbb{R}$ ($y_i = f(x_i) + \varepsilon$).

Find an estimator \hat{f} close to f .

The second definition is a convenient, simple formulation, when no explicit assumption or strict analysis needs or can be done, and a practical point of view is taken. It is not wrong or oversimplifying the problem, but can be misleading if taken too literally. The term *noise* somewhat implies that y is deterministically generated from x , and the data presents additional noise, coming from a measurement process or unobserved variables. This is not erroneous in the sense that a model can be written as:

$$Y = \mathbf{E}(Y|X) + (Y - \mathbf{E}(Y|X)) , \quad (1.3)$$

thus decomposing Y into a deterministic function of X and a random expression where, hopefully, the dependence on X is null, limited or of a simple form. It is null indeed when (X, Y) is normal, and for the *height* example, eq. (1.3) gives

$$Y = \mu + \rho(X - \mu) + \varepsilon \quad \text{where } \varepsilon \sim \mathcal{N}(0, (1 - \rho^2)\sigma^2) , \quad (1.4)$$

and the decomposition is fruitful, with a linear deterministic term and a normal random term independent of X and Y . The word *noise* may not be well suited, but the concept behind it of aggregating the randomness into a single variable Z that takes a simple form is generally much useful. However, the risks of sticking to this angle and forgetting the whole statistical model are twofold:

- considering that f can be an arbitrary function, or restraining its possible values by considerations that are detached from its inner statistical nature,
- neglecting the random part by wrongfully assuming a result as simple as eq. (1.4).

These risks are limited; in particular, the ubiquity of normal distributions makes the assumption of a normal “noise” with constant variance often justified. Nevertheless, they should be considered when designing or using estimator-computing algorithms. In the following, we take the general approach of the problem by definition (3), and discuss the need and form of choices, assumptions, restrictions, prior knowledge, regularization, all of which can be more or less derived from the statistical nature of the problem, and more or less arbitrary.

1.1.1.1 Choosing an estimator

Let us qualify as *compatible* an estimator from which the data *could* have been generated. When \mathcal{X} is not countable, it is not possible to get several observations of Y for all values of X , and regardless of the –countable– number of observations, there is an infinity of compatible models. Hence it is needed, independently of the data, to make more or less arbitrary choices in the form of rules and/or

assumptions to associate one model to a set of observations. Naturally, this is also valid when \mathcal{X} is countable and even in the limiting case of a singleton where the problem becomes the estimation of a scalar $\mu = E(Y)$. Strictly speaking, any estimation of μ is compatible with any data, and $\hat{\mu}$ can be chosen arbitrarily, that is without deriving it from any assumption or statement, or by maximum likelihood, or maximum a posteriori, based on some assumptions on the distribution of Y and the distribution of μ . However, regardless of the assumptions, a scalar estimator can be tested for consistency and bias, which give a formal meaning to the expression *close to*. The sample mean $\hat{\mu} = \frac{1}{n} \sum_{i=1}^n y_i$ can be proven to be an unbiased and –strongly– consistent estimator of μ , meaning the following:

$$\begin{aligned} \forall n, \mathbf{E}(\hat{\mu} - \mu) &= 0 \quad (\text{no bias}) \\ \mathbf{P}\left(\lim_{n \rightarrow \infty} (\hat{\mu} - \mu) = 0\right) &= 1 \quad (\text{strong consistency}) \end{aligned}$$

When, for example, $\mathcal{X} = \mathbb{R}$, and without any further assumption, the problem of finding an estimator for $\mathbf{E}(Y|X)$ is equivalent to a continuum of independent scalar problems, one for each possible value of X , and each of these problems have a probability one to have no observation at all. Then the closeness of \hat{f} to f requires arbitrary choices, reasonable assumptions, or explicit knowledge about f , to be formalized. Whatever form they take, they are prior considerations independent of the observations and result in defining a procedure (or *operator*) that maps these observations to one single model. This operator ultimately consists in more or less direct computations, and is generally defined by a numerical problem to be solved. Note that parts of an operator can be of an arbitrary nature (not derived from precise and valid prior knowledge about the process) but subject to analysis and validation under different assumptions. In other words, an operator can be designed either explicitly for working under precise assumptions, or from more practical considerations and statistical intuitions, possibly followed by analyzes of its validity for different families of problems.

1.1.2 Practical models

Indeed, a non-statistical criterion for guiding the choice of a model plays a strong role: it must be parametric –defined by a certain number of discrete or continuous parameters– in order to be manipulated at all, and must have a sufficiently simple form for the operator to be computed, and also for the practical use of the model afterwards.

Parameterizing \hat{f} necessarily restricts the space of possible models, and goes with the assumption that f belongs to this space or its neighbourhood. Its simplicity can take various forms and have different effects on the size of the model space, the complexity of computations, and its validity. The most common way of parameterizing is to use as a parameter a vector of real numbers, that

gives both expressiveness to the model, and possibilities of control and analysis over it.

The classical approach to regression is to define a parametric model that is reasonable in both terms of computational feasibility and capacity to approximate f , and define a measure to minimize, that reasonably approximates the discrepancy between \hat{f} and f , from the observations. The measure to minimize is referred to as a *loss function*, and together with the choice of the model space, it implies or is implied by a choice of which kinds of relationship can be well modelled.

This leads to the following more precise definition of the regression problem:

Definition 4 (parametric regression by optimization).

Let $f : \mathcal{X} \rightarrow \mathbb{R}$ be an unknown function, and $\mathcal{O} = (x_i, y_i)_{i=1, \dots, n}$ n observations generated by $y_i = f(x_i) + \varepsilon$, where ε is a random variable.

Let $\hat{f} : \Omega \times \mathcal{X} \rightarrow \mathbb{R}$ be a parametric estimator, $\hat{f}_\beta = \hat{f}(\beta, \cdot)$ an estimator with fixed parameter β , and $\mathcal{L} : 2^{\mathcal{X} \times \mathbb{R}} \times \Omega \rightarrow \mathbb{R}$ a loss function that approximates the discrepancy between f and a parametric estimator from the observations.

$$\underset{\beta \in \Omega}{\text{minimize}} \mathcal{L}(\mathcal{O}, \beta) \quad (1.5)$$

1.1.3 Linear regression

The practical side of this choice of model restrictions does matter much, and especially did in early days of the domain when computing resources were limited. That is one reason for the popularity of linear models, in which

$$\begin{aligned} \mathcal{X} &= \mathbb{R}^m \\ \Omega &= \mathbb{R}^m \\ \hat{f}_\beta(x) &= x^\top \beta = \beta_1 x_1 + \dots + \beta_m x_m \end{aligned}$$

A linear function usually includes a bias term, as β_{m+1} in

$$\hat{f}_\beta(x) = x^\top \beta = \beta_1 x_1 + \dots + \beta_m x_m + \beta_{m+1},$$

to overcome the restriction $\hat{f}(\mathbf{0}) = 0$, thus adding a degree of freedom. However, it is equivalent to adding a dimension to \mathcal{X} that shows a constant value, for example 1. Consider for example an initial predictor set $\mathcal{X}_0 = \mathbb{R}^k$. It can be mapped to $\mathbb{R}^k \times \{1\} \subset \mathbb{R}^{k+1}$, and by taking $m = k + 1$ and using the form defined above, we get

$$\hat{f}_\beta(x) = x^\top \beta = \beta_1 x_1 + \dots + \beta_k x_k + \beta_{m+1}.$$

Let us then, given the possibility of this pretreatment, limit the linear models to the bias-free, dot-product form. Not only is it convenient in terms of notations, but it also helps explaining, deriving, and implementing computations and algorithms. In this linear case, the parameter β may be referred to as a *coefficient vector*.

Another reason for using linear models, besides their simplicity, is that they are justified for a vast number of problems of interest. A lot of objects of measurement and experiment found in nature are known to exhibit such linear relationships, or quasi-linear ones. This remark yields two important general points.

The first one somewhat rephrases what was already mentioned but can help getting a clear view of the problematics. A regression experiment includes a first phase of identifying the general shape of the expected relation between X and Y , and more generally gather as much prior knowledge about it as possible, through the design of a parametric model. This restriction allows meaningful inferences from the *finite* number of samples. A second phase consists in the design of the loss function, in which another type of prior knowledge or assumptions must be accounted for: the joint distribution of (X, Y) , or in other terms where and how does randomness occur. This knowledge allows to approximate correctly the discrepancy between a model and f through *randomized* (or, from a different angle, *noisy*) samples. The third phase consists in having, choosing, or designing computational methods for minimizing the loss function. The feasibility of this phase is a necessary condition that interferes with the first two.

The second point raised by the fact that linear models are both simple to manipulate and often accurate, is that the two constraints involved in the designs of the estimators and loss, although of different natures and sometimes conflicting, are essentially compatible. The reason for this was mentioned above: the phenomena that the experimenter may study are always the results of simple interactions, and the simplicity is known to remain or emerge at higher levels of organization / observation, as exemplified by the Central Limit Theorem. In the *height* example, the function f was linear, as the result of (X, Y) being normally distributed, and more generally, linearity and normality are two interdependent natural forms of organization, that persist or emerge through additive processes.

1.1.3.1 Extended linear regression

Linear estimators are thus both simple to manipulate and optimize, and appropriate for many situations. Moreover, as will be illustrated below, their advantages in terms of optimizing the loss function is not related to their linearity in the predictor. The important property is the linearity in the optimized parameter β . Hence an estimator qualifies as linear from this sole property, and can proceed to arbitrary transformations of the predictor before a linear combination by the coefficient vector. These transformations are usually called *feature* functions, as they can be viewed as an alternative way to characterize predictors, by mapping them from the original description set \mathcal{X} to a new one, \mathbb{R}^p , on which a straight linear estimator is used: an extended linear estimator \hat{f} is defined by a set of feature functions

$$\phi_1, \dots, \phi_p : \mathcal{X} \rightarrow \mathbb{R},$$

forming a mapping function

$$\begin{aligned}\phi: \mathcal{X} &\rightarrow \mathcal{F} = \mathbb{R}^p \\ \mathbf{x} &\mapsto \phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \dots, \phi_p(\mathbf{x}))^\top\end{aligned}$$

and parameterized by a coefficient vector $\beta \in \mathbb{R}^p$:

$$\begin{aligned}\hat{f}_\beta: \mathcal{X} &\rightarrow \mathbb{R} \\ \mathbf{x} &\mapsto \phi(\mathbf{x})^\top \beta\end{aligned}$$

The set $\mathcal{F} = \mathbb{R}^p$ into which predictors are mapped is referred to as the *feature space*.

This extension has considerable advantages:

- it is inclusive of the first case of straight linear estimators;
- by definition, it can model nonlinear relationships between X and Y ;
- it can be applied to discrete data likes trees, diagrams, subsets;
- it can map the predictors into a large feature set, onto which a good linear estimator is more likely to be found.

The first point, if needed, can be clarified by the fact that in the case $\mathcal{X} = \mathbb{R}^m$, the mapping can be an identity, with $\phi(\mathbf{x}) = \mathbf{x}$, that is – with $\mathbf{x} = (x_1, \dots, x_m)^\top$

–

$$\phi_i(\mathbf{x}) = x_i, \quad \text{for } i = 1, \dots, m.$$

An example of the second point is the polynomial estimators. Polynomial functions are of a simple nature and likely to appear in the relationship between two variables. If $\mathcal{X} = \mathbb{R}^2$ and one restricts to a degree of three, the feature functions are $1, x_1, x_2, x_3, x_1^2, x_1x_2, x_1x_3, x_2^2, x_2x_3, x_3^2, x_1x_2x_3$.

The third point echoes what was mentioned at the beginning of this chapter about the usual definition of regression analysis: although it is said to study the relationship between one response variable and one or *several* predictor variables, the latter might as well be considered as one single predictor. If they are m scalars, they form one multivariate variable (vector) in \mathbb{R}^m , and more generally, a predictor variable need not be scalar or vector but can be of any nature, as long as the relationship can be analyzed, and in particular an estimator can be defined from \mathcal{X} to \mathcal{Y} . The nature of \mathcal{Y} is more critical, and we will keep concentrating here on $\mathcal{Y} = \mathbb{R}$, although other cases are possible, like function spaces (functional regression), or discrete sets (classification).

The last point is closely related to the second one. The example of polynomials illustrates that the feature mapping shatters $\mathcal{X} = \mathbb{R}^2$ into a richer space \mathbb{R}^{11} . This space is redundant for predictors, for they are fully described by components

2,3 and 4 (x_1, x_2 and x_3), but defines a much richer function space for linear estimators. The initial space of linear functions over \mathcal{X} is embedded in that of linear functions over \mathcal{F} , as a very “small” fraction. Ultimately, there always exists one linear estimator that exactly matches all observations if $p = n$ (assuming the observations are linearly independent in \mathcal{F}), and there exists an infinity of them for $p > n$. In the similar problem of *linear classification*, that aims at linearly *separating* two groups of data, this property was thoroughly studied in (Cover, 1965), and formulated by Covers’ theorem:

Theorem 1.1.1. *A complex pattern-classification problem, cast in a high-dimensional space non-linearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated.*

A popular way of featurization consists in using kernel functions. Such *kernel methods* include *Gaussian process regression* methods (GP), that use prior distributions and Bayesian inference, and *support vector machines* (SVM) that extend an optimal-separating-plane algorithm to such feature spaces. These methods are parameterized by a *Mercer* kernel function $k : \mathcal{X}^2 \rightarrow \mathbb{R}$, i.e. such that there exists a mapping function ϕ such that

$$\forall(x, x') \in \mathcal{X}^2, \quad \phi(x)^\top \phi(x') = k(x, x') \quad (1.6)$$

Although it is not necessarily the case, ϕ can generally be defined as associating a different feature function to every point of \mathcal{X} , yielding the following set of feature functions:

$$\mathcal{D} = \{\phi : \mathcal{X} \rightarrow \mathbb{R}, x \mapsto \phi(x) = k(c, x) \mid c \in \mathcal{X}\}$$

This set is isomorphic to \mathcal{X} , hence usually infinite, which makes the feature space infinite-dimensional and the set of linear estimators over it very rich. This richness is relative, however, because it is defined, and biased, by the choice of a single kernel function. The mapping function ϕ , previously defined by $\phi(x) = (\phi_1(x), \dots, \phi_m(x))^\top$ for a finite set of m features, now maps x to a point with as many attributes as there are points in \mathcal{X} , that could be noted $\phi(x) = (\dots k(x, c) \dots)_{c \in \mathcal{X}}$, and can be seen as a function $k(x, \cdot)$. The obvious drawback is that linear estimators are sums of infinitely many terms, which is not practical. The workaround is to define an estimator of which the computation involves only a finite number of dot products such as that in eq. (1.6), generally between observation and prediction points. This allows not to operate actual computations in the feature space, and necessarily implies that the result itself is not infinite-dimensional, that is only a finite subset of the features (those appearing in the dot products) is actually considered – which is desirable to obtain a practical estimator. This selection among the infinite number of features is preliminary, in the sense that it does not take the values of the observations into account. It is based on the *representer theorem* (SVM), or on conditioning on the observations (GP). Extensive reviews of GP and SVM –and related– methods

can be found in, respectively, (Rasmussen and Williams, 2006) and (Scholkopf and Smola, 2001).

With great power of expression comes great responsibilities of selecting the right model; that is, by considering a rich function space for possible estimators, finding one close to the real mean $E(Y|X)$, or, with a more practical point of view, that predicts well at unobserved points, can become a harder task. In particular, when the number of features is greater than the number of observations, an infinity of estimators associate the exact observed responses y_i to the observed predictors x_i , and uncarefully designed loss functions or procedures may select or tend to one of those, fitting exactly the observations without taking the random term into consideration (*overfitting*). More generally, when a linear estimator is used, based on well grounded statistical considerations, like in the *height* example, the loss function can be well defined, on the same considerations, and yield a correct estimator, but the same loss applied to an artificially rich function space can yield erroneous results.

1.1.4 Loss functions

This well-defined loss function for straight linear regression is the squared residual, yielding the method of least-squares. The method was actually designed first, empirically, by Carl Friedrich Gauss, before its optimality for linear estimators was proven through the Gauss-Markov theorem. This theorem states that the least-squares estimator has the lowest possible variance among unbiased linear estimators of a linear function. Let us give here an intuitive derivation of the method and explanation of its optimality.

Lets us first consider the estimation of the mean μ of a simple scalar random variable Y . We know that the empirical mean $\bar{\mu} = \frac{1}{n} \sum_{i=1}^n y_i$ is a good estimator, being unbiased and strongly consistent. μ and $\bar{\mu}$ can be alternatively defined from the following:

The mean is the minimizer of the variance.

This means that if a variable m is defined as a “candidate” mean, and one considers the variance of Y around m with $V(m) = E((Y - m)^2)$, we have

$$\mu = \arg \min_m V(m)$$

i.e.

$$E(Y) = \arg \min_m E((Y - m)^2)$$

and similarly,

$$\bar{\mu} = \arg \min_m \bar{V}(m)$$

i.e.

$$\frac{1}{n} \sum_{i=1}^n y_i = \arg \min_m \frac{1}{n} \sum_{i=1}^n (y_i - m)^2$$

This can be shown easily and allows to define this estimator as the unique minimizer of a quadratic –loss– function.

In the regression setting, the Gauss-Markov assumption is that there exists a coefficient $\boldsymbol{\beta}$ such that the “error” or “noise” term defined for a realization (X_i, Y_i) of (X, Y) by

$$\varepsilon_i = Y_i - X_i^\top \boldsymbol{\beta}$$

verifies

$$\begin{cases} \mathbb{E}(\varepsilon_i) = 0 \\ \mathbb{E}(\varepsilon_i^2) = \sigma^2 < \infty \\ \mathbb{E}(\varepsilon_i \varepsilon_j) = 0 \end{cases}$$

It essentially means that (X, Y) can be seen as drawn by first sampling X , then obtaining Y by the sum of the linear term and a noise term independent of X . This independency implies, informally, that each observation can be transformed, somewhat concentrated, into an observation of that scalar noise term, and that the estimation of $\boldsymbol{\beta}$ can be concentrated on the estimation of the mean of ε (zero), by the following:

let $\hat{\boldsymbol{\beta}}$ be a candidate value for the unknown parameter $\boldsymbol{\beta}$, and let us write

$$\hat{\boldsymbol{\beta}} = \boldsymbol{\beta} + \boldsymbol{\Delta}_\beta$$

$$\begin{aligned} 0 &= \arg \min_m \mathbb{E}((\varepsilon - m)^2) \\ \implies 0 &= \arg \min_m \mathbb{E}((Y - X^\top \boldsymbol{\beta} - m)^2) \\ \implies \mathbf{0} &= \arg \min_{\boldsymbol{\Delta}_\beta} \mathbb{E}((Y - X^\top \boldsymbol{\beta} - X^\top \boldsymbol{\Delta}_\beta)^2) \\ \implies \boldsymbol{\beta} &= \arg \min_{\hat{\boldsymbol{\beta}}} \mathbb{E}((Y - X^\top \hat{\boldsymbol{\beta}})^2) \end{aligned}$$

and similarly, the fact that

$$\bar{\varepsilon} = \arg \min_m \frac{1}{n} \sum_i \varepsilon_i$$

is a good estimator of 0 implies the same properties for

$$\begin{aligned}
 \bar{\beta} &= \arg \min_{\beta} \frac{1}{n} \sum_i (y_i - \mathbf{x}_i^\top \beta)^2 \\
 &= \arg \min_{\beta} \sum_i (y_i - \mathbf{x}_i^\top \beta)^2 \\
 &= \arg \min_{\beta} \|\mathbf{y} - \mathbf{X}^\top \beta\|_2^2 \quad (\text{vector notation})
 \end{aligned} \tag{1.7}$$

as the convergence of the latter is directly linked to the convergence of $\bar{\varepsilon}$ to 0.

The solution of the least-squares equation (1.7) is given by zeroing the gradient of the squared residual wrt. β :

$$\begin{aligned}
 \nabla_{\bar{\beta}} \|\mathbf{y} - \mathbf{X}^\top \bar{\beta}\|_2^2 = 0 &\iff \mathbf{X}(\mathbf{y} - \mathbf{X}\bar{\beta}^\top) = 0 \\
 &\iff \mathbf{X}\mathbf{y} = \mathbf{X}\mathbf{X}^\top \bar{\beta} \\
 &\iff \bar{\beta} = (\mathbf{X}\mathbf{X}^\top)^{-1} \mathbf{X}\mathbf{y}
 \end{aligned} \tag{1.8}$$

$\mathbf{X}\mathbf{X}^\top$ is referred to as the Gram matrix of the observations, that concentrates the n observations of X into a $p \times p$ matrix of correlation between the p parameters. The term correlation is taken here in its largest acceptance, but can be applied more formally when the parameter is taken as a random variable, as in Gaussian process methods. It is positive-definite, thus invertible and allowing the solution 1.8, if $p \geq n$ – after removing possible degeneracies (linear dependencies) in the set of features.

A typical example of a valid Gauss-Markov assumption is when it is derived from a normality assumption such as the one in the *height* example. We have seen in eq. (1.4) that under the joint-normality assumption, Y is linear in X with an error term that is normal with a fixed variance.

The above explanation was intended to give an intuition, but the Gauss-Markov theorem can be proven more formally by showing that the least-squares estimator has the lowest variance among unbiased linear estimators. The bias and variance are defined here on the estimated parameter with respect to the randomness of the observation set, through the following decomposition:

$$\underbrace{\mathbb{E} \left(\|\hat{f} - f\|_2^2 \right)}_{\text{MSE}} = \underbrace{\|\mathbb{E} \left(\hat{f} \right) - f\|_2^2}_{\text{bias}} + \underbrace{\mathbb{E} \left(\|\hat{f} - \mathbb{E} \left(\hat{f} \right)\|_2^2 \right)}_{\text{variance}} \tag{1.9}$$

where the mean squared error (MSE) expresses the expected distance between the estimator and the true function, the bias is minimized if the estimator can and does, in expectation, get close to the true value, and the variance expresses how far the estimator can get from its expected value depending on the observations.

If the linearity assumption of f is valid and \hat{f} is linear, it is biased only if its expectation does not match f , although it could. Conversely, if a linear estimator is as good as possible in terms of expected value, but f is not linear, the bias expresses the distance between the true function and the set of linear estimators. If the true function is not known to be linear in the original description space of X , then mapping it to a higher-dimensional feature space and using a least-squares linear estimator reduces the bias because the linearity assumption becomes more valid, and even totally valid for sufficiently large and regular spaces. But the capacity to estimate well f by a linear function comes at the price of an increased variance of the estimator, from the following simple fact: a greater number of scalar parameters to estimate requires more observations to be well estimated; if the number of observations m is lower than the dimension of the feature set p , the least-squares parameter is not even properly defined, as there is an infinity of parameters, isomorphic to \mathbb{R}^{n-p} , that minimize the squared residual. Even if one is chosen among them from one criterion or another, it yields a residual of 0, that is the estimator fits exactly the observations.

Hence, when f is not known to be linear in a small number of known features, a “blind” use of, or projection into, a high-dimensional feature space, and the use of a least-squares linear estimator, although asymptotically correct, is not a good choice, because a large number of observations is needed, and an estimator based on too few samples is “biased” (in the general meaning of the term) towards overfitting. Thus, additional assumptions or restrictions are needed to guide the estimator to more plausible solutions, introducing a bias for the benefit of a lower variance. This approach is often referred to as *regularization* of the least-squares loss function.

1.1.5 Regularization

The term *regularization* can be used when some form of regularity is required for the estimator, in addition to an initial criterion – in the case discussed here, being linear and minimizing the squared residual. The first well-defined and powerful form of regularization appeared with *ridge regression*. The initial motivation was twofold: reduce instability in the algebraic computations of least-squares, and reduce the mean squared error, as explained above, by introducing a small bias in the hope of a substantially lower variance. The latter point was addressed partly empirically and partly from intuitions driven by theory, and both yielded the solution of adding a small diagonal term to the Gram matrix in the least-squares formula:

$$\bar{\beta} = (\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I})^{-1} \mathbf{X}\mathbf{y}, \quad (1.10)$$

The method and its numerous properties were then stated and analyzed formally in (Hoerl and Kennard, 1970a) by its two independent authors.

For one of the authors, eq. (1.10) came as a result of bounding the parameter by imposing the following constraint:

$$\sum_i \beta_i \leq t \quad (1.11)$$

for some hyper-parameter t . The regularization effect is twofold, or can be seen from two points of view: it prevents low eigen values of the Gram matrix and afferent computational issues, which is a regularization of the matrix itself, and, somewhat equivalently, it prevents big variations in the function, which restricts the space of possible estimators to “smooth” ones, which is a regularization of the estimators.

The derivation of eq. (1.10) from eq. (1.11) will not be recalled here, but another equivalence can be seen in a few lines:

$$\begin{aligned} \bar{\beta} = (\mathbf{X}\mathbf{X}^\top + \lambda\mathbf{I})^{-1} \mathbf{X}\mathbf{y} &\iff \mathbf{X}\mathbf{X}^\top \bar{\beta} - \mathbf{X}\mathbf{y} + \lambda\bar{\beta} = 0 \\ &\iff -\mathbf{X}(\mathbf{y} - \mathbf{X}^\top \bar{\beta}) + \lambda\bar{\beta} = 0 \end{aligned}$$

and by integration

$$\bar{\beta} = (\mathbf{X}\mathbf{X}^\top + \lambda\mathbf{I})^{-1} \mathbf{X}\mathbf{y} \iff \bar{\beta} = \arg \min_{\beta} \frac{1}{2} \|\mathbf{y} - \mathbf{X}^\top \beta\|_2^2 + \frac{1}{2} \lambda \|\beta\|_2^2$$

This results in a formulation where the loss function itself is regularized to achieve a trade-off between the minimization of the residual and that of the ℓ^2 norm of the coefficient. Thus there are equivalent formulations of a *constrained least-squares* problem

$$\underset{\beta}{\text{minimize}} \|\mathbf{y} - \mathbf{X}^\top \beta\|_2^2 \quad \text{subject to } \|\beta\|_2^2 \leq t \quad (1.12)$$

and a *regularized least-squares* problem

$$\underset{\beta}{\text{minimize}} \|\mathbf{y} - \mathbf{X}^\top \beta\|_2^2 + \lambda \|\beta\|_2^2 \quad (1.13)$$

with solution

$$\beta = (\mathbf{X}\mathbf{X}^\top + \lambda\mathbf{I})^{-1} \mathbf{X}\mathbf{y} \quad (1.14)$$

1.2 The Least Absolute Shrinkage Operator

Another way to regularization consists in variable selection, that is computing the least-squares estimator on only a subset of the variables, qualified as *active*. The original motivation was not so much to find a better bias-variance compromise, as to provide a simpler model, best suited for interpretation or manual computations, avoiding the unnecessary use of variables with a small contribution, that

can be assimilated to noise. The problem of finding the best subset of a given size k in the sense of minimizing the squared residual or a similar loss function is known to be NP-complete, that is all possible subsets must be considered and the corresponding estimators computed. This aspect and the motivations correspond to the setting of straight linear regression, where the number of original, natural predictors is already relatively small.

In this framework, the research first concentrated on finding alternative criteria to least-squares for choosing the best subset. A noticeable result was Colin Mallows' C_p criterion in (Mallows, 1973), that is in essence a normality test on the residual. Using such a criterion yields a good bias, resulting in the following regularization procedure:

- enlarge the set of possible estimators, from the unique least-squares one to the partial least-squares on all subset of variables;
- choose among them from a criterion that, unlike the least-squares, takes the noise into consideration and is not biased towards overfitting.

This procedure crosses two different criteria by a first selection based on one, followed by a choice based on another one.

The effect of variable selection in itself on overfitting can be understood by the following: if a variable is used that is only negligibly correlated to the response, this variable is as much correlated to the noise as to the mean response, and will be “used” as such by the least-squares procedure. Thus the presence of such variables gives degrees of freedom to the least-squares to fit the noise. When they are numerous, the simple fact of bounding the number of selected variables and select among them by the least-squares criterion can prevent overfitting. We have mentioned that best-subset problem is NP-complete, and that is why some alternative but similar selection procedures were developed.

The first noteworthy one, called *stepwise regression*, consists in selecting or eliminating variables in a sequential greedy fashion, that is by considering the immediate, local effect of these actions. It was first designed and used by many practitioners, the first written account appearing in (Draper and Smith, 1966). The natural way to proceed when only a few variables need be eliminated is through *backward elimination*: starting from the full least-squares solution, one estimates the contribution of each variable by some statistical test and removes the least important one. The partial least-squares solution is then computed on the reduced set and the operation repeated. The opposite procedure, called *forward selection*, consists in starting from the null solution and sequentially select the most *locally* important variable, *locally* referring to the sequence of steps, that is these steps design a sequence of estimators \hat{f}_i , each step selecting the most useful variable to reduce the current remaining residual $\mathbf{y} - \hat{f}_i(\mathbf{x})$. In this case the least-squares criterion itself can be used, the contribution of each variable to the squared residual being assessed by the derivative of the latter with respect to the formers. This approach can be qualified as greedy, because

the addition of the most locally useful variables is not equivalent to the most useful set of variables, especially given that these variables are not considered for removal along the sequence. This method is satisfying when a few variables need be selected, since no great drift (discrepancy between local and global benefit) can occur in a few steps. These two approaches can however, given their greedy nature, give unoptimal results when a good and sparse estimator is to be found in a middle range of number of active variables, like 20 to 80 out of 100.

Another alternative to best-subset selection was proposed in (Tibshirani, 1996). It consists in replacing a constraint on the number of active variables by a constraint on the sum of the absolute value of the coefficients. This expression is usually referred to as the ℓ^1 norm of the coefficient vector. The constraint, added to the least-squares criterion, gives the following Least Absolute Shrinkage and Selection Operator (LASSO):

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \sum_{i=1}^n y_i - \mathbf{x}_i^T \boldsymbol{\beta} \quad \text{subject to} \quad \sum_{i=1}^m |\beta_i| \leq t$$

or equivalently

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \|\mathbf{y} - \mathbf{X}^T \boldsymbol{\beta}\|_2^2 \quad \text{subject to} \quad \|\boldsymbol{\beta}\|_1 \leq t \quad (1.15)$$

The term *operator* is generally used to designate a function of which the value cannot be obtained by straightforward computation, from an analytical formula, but requires an algorithmic approach. This is the case here, since an estimator is a function of the observations and possibly of some parameters, that has an analytical formulation like (1.8) in the least-squares case, or (1.14) for ridge regression, but no closed-form expression of the LASSO estimator can be given.

A motivation for bounding the ℓ^1 -norm is that the number of nonzero coefficients, which is the bounded variable in best subset selection, is sometimes referred to as the ℓ^0 -norm. Although not a true norm, it is the limit of ℓ^α -norm when α tends to zero : for a vector \mathbf{v} of \mathbb{R}^n ,

$$\sum_{i=1}^n \mathbf{1}_{v_i \neq 0} = \lim_{\alpha \rightarrow 0} \sum_{i=1}^n |v_i|^\alpha,$$

where $\mathbf{1}_a$ equals one when assertion a is true and zero when it is false.

The ℓ^1 -norm is the closest norm to ℓ^0 to be convex, thus making an interesting substitute that combines sparsity properties and computational feasibility. The relations between the norms, and the sparsity and convexity properties are illustrated in figures 1.2 and 1.3.

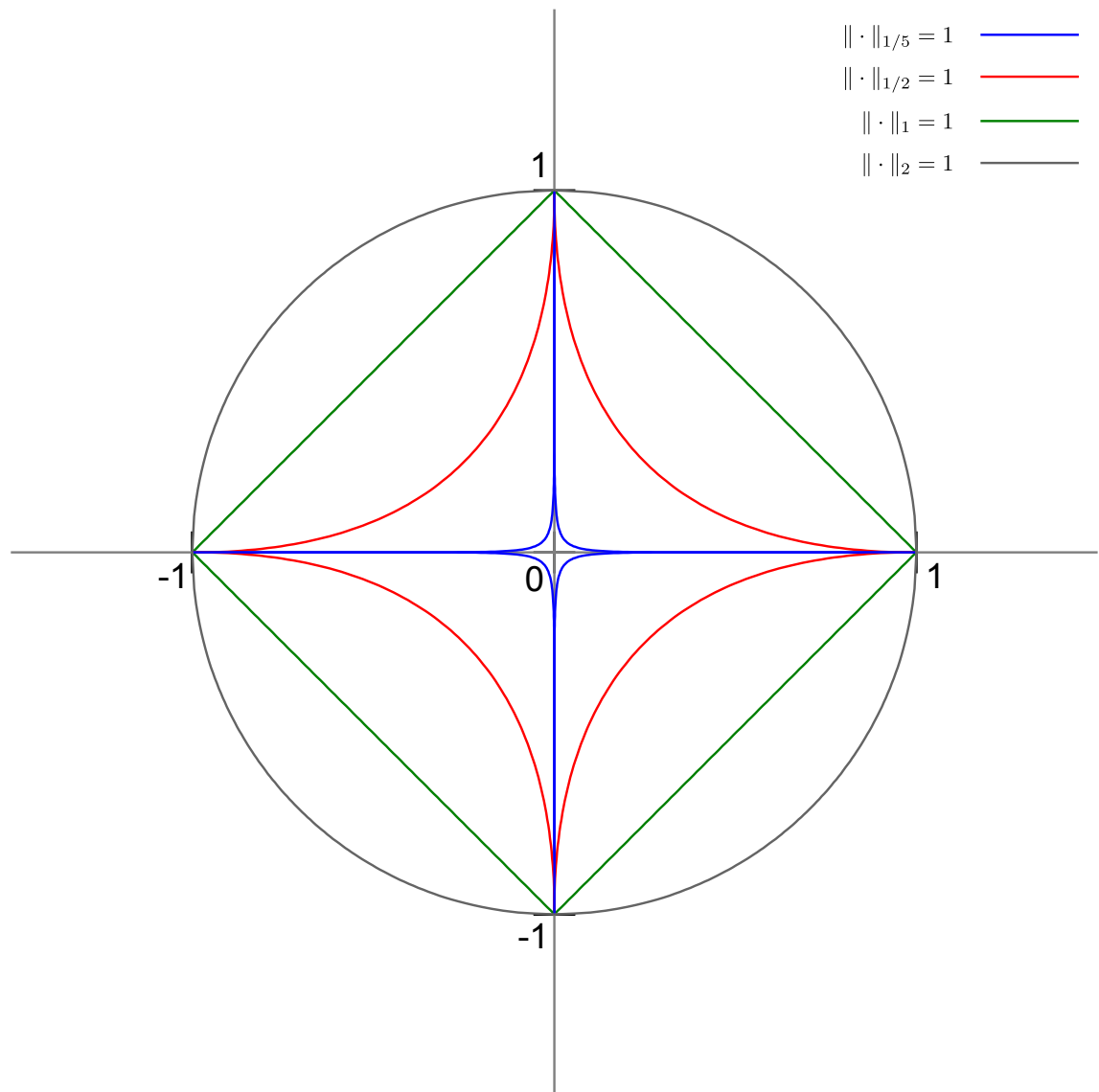


Figure 1.2: Different ℓ^q -norms in \mathbb{R}^2 , for $q = 1/5, 1/2, 1, 2$; ℓ^1 is the first convex one (with q increasing), and also the last one with an “angular” property of presenting vertices where one or several attributes are zero.

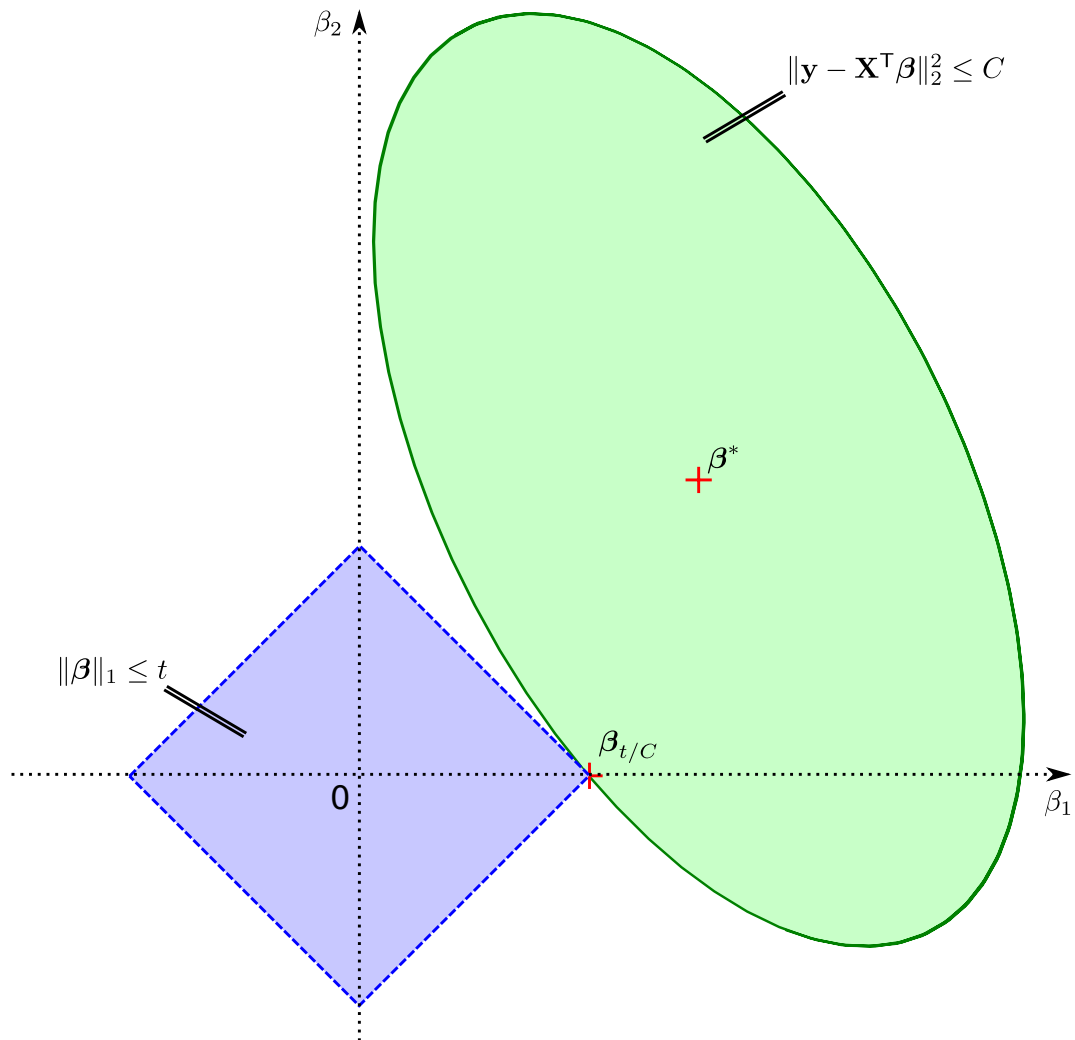


Figure 1.3: Interaction of the ℓ^2 -norm of the residual (loss) and ℓ^1 -norm of the coefficient, in the coefficient-space view.

Their convexities (assuming the loss is strictly convex) make both the minimizer of the loss under the ℓ^1 -norm constraint, and the minimizer of the norm under the quadratic loss constraint, unique and lying on the surfaces of the constraints, as long as one does not include the centre of the other. This reciprocal unicity defines a bijection between corresponding values of t and C . The point $\beta_{t/C}$ is the common solution to both problems of this example. Given the angular property of the ℓ^1 -norm, the smaller the constraint t , the more likely $\beta_{t/C}$ is to lie on one of its vertices.

The closeness of ℓ^1 and ℓ^0 regularizations can be studied through the ability of ℓ^1 -regularized estimators to recover the non-zero variables in a sparse model to which noise was added. In (Donoho and Stodden, 2006), through geometrical analysis and experimental illustration, a phase transition phenomenon was identified between settings for which this recovery is efficient, and complementary settings for which the ℓ^0 minimization remains a hard (combinatorial) task. The settings on which this transition is identified are the sparsity of the original model (or signal), and the number of observations, both relatively to the number of variables to select from. A subsequent work in (Xu et al., 2010) showed $\ell^{1/2}$ regularization to be a good compromise between the recovery property of ℓ^0 and the feasibility of ℓ^1 , as the recovery improves only slightly for $q < 1/2$, while $\ell^{1/2}$ regularization remains reasonably feasible, by an algorithm they proposed in (Xu et al., 2010), that solves a sequence of re-weighted ℓ^1 -regularized problems (the penalization factors \mathbf{w} in the LASSO definition given in section 1.2.2 are updated at each step).

1.2.1 Algorithms for solving the LASSO

Problems similar to the LASSO were studied before, like least-squares under linear constraints or quadratic programming. Hence, together with the definition of the problem, two algorithms were proposed in (Tibshirani, 1996) that were derived from methods defined for these problems. The first one was adapted from the Non-Negative Least-Squares (NNLS) method defined in (Lawson and Hanson, 1974), that solves, as its name suggests least-squares problems under non-negativity constraints, and equivalently under arbitrary linear constraints. The adaptation is somewhat confusing; it is based on the statement that the number of constraints induced by the ℓ^1 -norm one is 2^p (if p is the number of variables), that is the variables of the NNLS are all possible subsets of the regression variables. The second proposition, that is said to be a completely different approach, is to formulate the LASSO as a quadratic program (QP) and use standard QP algorithms. The summary of these two methods is that one has p variables and 2^p constraints, whereas the second has $2p$ variables and $2p + 1$ constraints. The derivations of algorithms made throughout chapter 2 should clarify that the application of NNLS to the LASSO is more straightforward and involves only p or $2p$ variables.

The following noticeable proposition of LASSO algorithms was made in (Osborne et al., 2000a), and the first method is actually equivalent to a proper adaptation of NNLS to the LASSO, as will be shown in chapter 2; this algorithm was called the *active set* method. The second one, called the *homotopy* method, allows to compute the LASSO estimators for all meaningful values of the constraint parameter t , with little extra cost.

The third important publication was (Efron et al., 2004), in which the connection between an improved form of forward selection, forward stagewise regression, and the homotopy method is made, through the *LARS* algorithm. This caused a

growing interest in the LASSO, and the LARS algorithm is now usually known, used and cited for its LASSO modification, which is essentially the homotopy method, and the homotopy method known as the LARS algorithm.

One of the reasons for the success of the LARS formulation is probably that it addresses the alternative *regularized* formulation of the LASSO. Indeed, just as for ridge regression, there is an equivalence between eq. (1.15) and the following:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \|\mathbf{y} - \mathbf{X}^T \boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 \quad (1.16)$$

The equivalence between the problems (1.16) and (1.15) is often mentioned in the literature about the LASSO, but – as it seems – never explicited, that is given a proof and/or a characterization of the relation between the two regularization parameters. The two problems are even sometimes said to be simply “closely related” rather than equivalent. This equivalence cannot be derived as simply as was done for ridge regression in section 1.1.5, but can be seen as a rather simple application of Lagrange multipliers for convex optimization. However, rather than exposing this here, we characterize this relation in section 2.2.2, after the value of λ associated to a given t emerges from the derivation of an algorithm made in section 2.2.2.

An alternative approach for solving the LASSO appeared in (Fu, 1998), and under different forms in other publications, but raised a real interest only recently after it was presented and generalized in (Friedman et al., 2007). It consists in using a simple and general optimization technique used by practitioners but subject of only a few theoretical publications, and generally referred to as *coordinate descent*. The algorithm is explained in section 2.5, and further discussed.

The active set method and the equivalent NNLS algorithm did not raise a great interest for various reasons, among which a misunderstanding of the problems equivalence, and the fact that it addresses the constrained formulation (1.15). The next chapter aims at giving a general form of this algorithm and applying it to the LASSO as simply and clearly as possible. A prior step made in the following is to explicit the relation of the LASSO to quadratic programming, and more generally how the ℓ^1 norm of the coefficient can be translated to be handled more easily. Prior to this, let us reformulate and generalize the LASSO problem.

1.2.2 Extended definition

The first reformulation concerns the distinction between straight linear regression on measured variables and extended linear regression in a feature space. As pointed out, no such distinction need be done, and we take the following approach: the predictor X belongs to a set \mathcal{X} that need not be specified or be of a numerical nature, and is described by p numerical features obtained from a set of feature functions $\sigma = \{\phi_1, \dots, \phi_p\} \subset \mathbb{R}^{\mathcal{X}}$. In other words, no distinction is made between natural measurements or characteristics, and mathematical functions

applied to them; they are all feature functions from \mathcal{X} to \mathbb{R} . The value of $\mathbf{x} \in \mathcal{X}$ in the feature space is given by

$$\boldsymbol{\phi}(\mathbf{x}) = (\phi_1(\mathbf{x}), \dots, \phi_p(\mathbf{x}))^\top,$$

and the descriptions of the n observations of X are aggregated in

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \phi(\mathbf{x}_1) & \dots & \phi(\mathbf{x}_n) \\ | & & | \end{pmatrix} = \begin{pmatrix} - & \boldsymbol{\phi}_1(\mathbf{x}) & - \\ & \vdots & \\ - & \boldsymbol{\phi}_p(\mathbf{x}) & - \end{pmatrix} = \begin{pmatrix} \phi_1(\mathbf{x}_1) & \dots & \phi_1(\mathbf{x}_n) \\ \vdots & & \vdots \\ \phi_p(\mathbf{x}_1) & \dots & \phi_p(\mathbf{x}_n) \end{pmatrix}$$

The second change made in the definition of the LASSO is related to a point not mentioned before: the normalization of the features. Indeed, when selecting features by penalizing their coefficients, it is natural to ensure that these coefficients are comparable, in the sense that they have similar ranges of effect on the estimator. This means normalizing the features, that is scaling them by the inverse of their norm:

$$\phi \leftarrow \frac{1}{\|\phi\|_2} \phi$$

The exact norm of a feature is not generally accessible; it is given by

$$\|\phi\|_2^2 = \int_{\mathcal{X}} \phi(X)^2 dX,$$

which implies having an analytical form of ϕ , being able to integrate its square, and knowing the probability distribution of X . Hence the following empirical estimation may be used:

$$\|\phi\|_2^2 \approx \|\boldsymbol{\phi}(\mathbf{x})\|_2^2 = \sum_{i=1}^n \phi(\mathbf{x}_i)^2$$

This estimation is even justified in itself by the fact that the effects of each feature to the *observations* are compared, thus they should be normalized with respect to these observations. Yet an argument for using the real norm or a closer approximation is that the effects on unobserved points should and can be taken into account to limit the variance. Another point is that one does not necessarily want a “fair” comparison of features, but may *a priori* favourize or penalize them differently, from a prior belief that some have better generalization properties. Whatever form of normalization is chosen and differentiated penalization is made, it is worth including it explicitly in the definition of the LASSO, rather than just mentioning the need or possibility of a prior rescaling of the features. It can be shown straightforwardly that scaling a feature by a division is equivalent to multiply its coefficient in the penalization term, by the same factor. Thus we add to the definition of a LASSO problem a penalization factor w associated to each feature.

A third change, made possible by using a notion of feature functions rather than variables, acknowledges the fact that since a selection of a finite number of features is operated by the ℓ^1 constraint, there is no need to restrict to a finite number of “candidate” features, which is implied by using the matricial form of (1.15) or (1.16). Problems may occur in the feasibility of algorithms in non-finite cases, but this does not affect a proper definition of the operator, which is the following, with the three differences that were just mentioned:

Definition: LASSO *Given*

- n observations $(\mathbf{x}_i, y_i)_{1 \leq i \leq n}$ of joint variables $(X, Y) \in \mathcal{X} \times \mathbb{R}$,
- a set of feature functions $\mathcal{D} \subset \mathbb{R}^{\mathcal{X}}$,
- a feature penalization function $w : \mathcal{D} \rightarrow \mathbb{R}_+$,
- a constraint $t \in \mathbb{R}_+$ or a regularization parameter $\lambda \in \mathbb{R}_+$,

the LASSO associates to the observations an estimator defined as a linear combination of features from \mathcal{D} that minimizes the squared residual subject to a weighted ℓ^1 -norm constraint on the linear coefficient vector:

$$\underset{(\sigma, \beta) \in 2^{\mathcal{D}} \times \mathbb{R}}{\text{minimize}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \beta\|_2^2 \quad \text{subject to } \|\beta\|_{\mathbf{w}_\sigma} \leq t, \quad (1.17)$$

or the squared residual with a corresponding additive regularization term:

$$\underset{(\sigma, \beta) \in 2^{\mathcal{D}} \times \mathbb{R}}{\text{minimize}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \beta\|_2^2 + \lambda \|\beta\|_{\mathbf{w}_\sigma}, \quad (1.18)$$

where σ , referred to as the active set, is a finite subset of features, arbitrarily ordered: $\sigma = \{\phi_1, \dots, \phi_k\} \subset \mathcal{D}$, β is an associated coefficient vector, of which no component is zero,

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \quad \mathbf{w}_\sigma = \begin{pmatrix} w(\phi_1) \\ \vdots \\ w(\phi_n) \end{pmatrix},$$

$$\mathbf{X}_\sigma = \begin{pmatrix} - & \phi_1(\mathbf{x}) & - \\ & \vdots & \\ - & \phi_k(\mathbf{x}) & - \end{pmatrix} = \begin{pmatrix} \phi_1(x_1) & \dots & \phi_1(x_n) \\ \vdots & & \vdots \\ \phi_k(x_1) & \dots & \phi_k(x_n) \end{pmatrix},$$

and $\|\cdot\|_{\mathbf{w}}$ denotes a weighted ℓ^1 norm:

$$\|\mathbf{v}\|_{\mathbf{w}} = \sum_i w_i |v_i|$$

1.2.3 LASSO programming

The specificity of the ℓ^1 norm, that gives both its selection property and the impossibility to derive a closed-form solution to the LASSO, is its non-differentiability at points where a coefficient is zero. However, if a coefficient is constrained to have a given sign, the ℓ^1 norm is differentiable with respect to that coefficient, or, equivalently, the derivative with respect to a coefficient is defined everywhere but at points where this coefficient is zero:

$$\beta_i \geq 0 \implies \frac{\partial \|\boldsymbol{\beta}\|_{\mathbf{w}}}{\partial \beta_i} = w_i$$

$$\frac{\partial \|\boldsymbol{\beta}\|_{\mathbf{w}}}{\partial \beta_i} = \begin{cases} w_i & \text{if } \beta_i > 0 \\ \text{undefined} & \text{if } \beta_i = 0 \\ -w_i & \text{if } \beta_i < 0 \end{cases}$$

This gives two equivalent ways of handling this norm. The first can be called the *positivity trick* and consists in doubling the set of features by adding the negative counterpart of each of them:

$$\mathcal{D} \leftarrow \mathcal{D} \cup \{-\phi \mid \phi \in \mathcal{D}\}$$

This allows to impose a positivity constraint on the coefficients, since $-\beta\phi$ can be replaced by $\beta(-\phi)$. Under this positivity condition, the ℓ^1 norm can be written as a simple linear expression:

$$\|\boldsymbol{\beta}\|_{\mathbf{w}} = \sum_i w_i |\beta_i| = \sum_i w_i \beta_i = \mathbf{w}^T \boldsymbol{\beta}$$

Whence, the ℓ^1 norm constraint can be decomposed into positivity constraints and what we shall call an *oblique* constraint:

$$\boldsymbol{\beta} \geq \mathbf{0} \tag{1.19}$$

$$\mathbf{w}^T \boldsymbol{\beta} \leq t \tag{1.20}$$

at the price of doubling the set of features. This results in a quadratic program: minimize a quadratic function function of $\boldsymbol{\beta}$ under linear constraints. However, a direct application of a general QP solver is not a good solution, for two reasons:

- there is a unique oblique constraint, and rather than defining an additional slack variable, it can be handled explicitly, as will be seen in section 2.1;
- the number of variables (features) is only artificially doubled, and an algorithm can easily factorize the computations regarding one feature and its negative counterpart.

An alternative and equivalent way of handling the ℓ^1 norm is by *monitoring* the coefficients signs: a feature may be selected together with a condition on the sign of its coefficient, that allows to differentiate $\|\beta\|_{\mathbf{w}}$. When an algorithm is navigating in the coefficient space in search of an optimal solution, it should then monitor the signs assumptions, to unselect a feature when it is violating the assumption under which it was selected. This *sign monitoring* view is more cumbersome and related to characteristics of an algorithm. Therefore, we shall use the *positivity trick* in the following chapter, that exposes the principles of the active set method and its application, from the general case to LASSO programs. The *sign monitoring* view will be used when exposing the precise and implementable active set and homotopy algorithms for the LASSO.

Active set Algorithms

In this chapter, we derive the general active set descent method and its application to LASSO, in either constrained or regularized formulation, and derive from there the homotopy method. Although these two methods have already been presented in the literature, even twice for the second one, this chapter yields several contributions. The first one is to give a simple and very general formulation of the active set method, that is surprisingly absent from the literature. This formulation yields a clear and simple application to the LASSO, that may be more easily understood than those of (Osborne et al., 2000a) and (Osborne et al., 2000b). One evidence of this is that the authors themselves did not realize that the method applies directly to the regularized formulation of the LASSO, and proposed instead to repeatedly switch from the regularized to the constrained formulations, proceeding each time to a whole run of their algorithm before adjusting the constraint. Our formulation makes it clear that the active set method applies, even more naturally, to the regularized version, and results in an even simpler algorithm. This chapter also clarifies the equivalence relation between the two LASSO formulations, which can be derived from the general duality theory, but appears here in a practical and natural fashion, for the specific case of the LASSO. The presentation of the homotopy method is also original by being derived from the active set descent method, and using different variables, aiming at a simpler understanding and formulation of the algorithm. These two points also allow to handle a degenerative case, which is made in the following chapter.

2.1 Solving the constrained problem on a given active set

The selection property of the constraint is naturally the source of both benefits and difficulties of ℓ^1 regularization. If this part of the problem is left aside, that is if the active set of a LASSO solution – that we may refer to as an *optimal* active set – is assumed to be known, the remaining task reduces to the straightforward computation of minimizing a quadratic function under a linear constraint.

The algorithms presented in this chapter proceed by repeatedly performing this computation for successive active sets that are either assumed to be optimal. The sequences of active sets are constructed by successive inclusions and removals of one element, following considerations of –respectively– *sufficiency* and *necessity/feasibility*.

In this section, we detail this computation of a linearly constrained least-squares problem, from which the notions of feasibility and optimality appear naturally. This step also involves a Lagrange multiplier that leads to the regularized version of the problem and enlightens the relation between these two formulations.

2.1.1 Optimality of an active set

Let us decompose the optimization problem into the sub-problem of finding an optimal active set and that of computing its coefficient vector.

In the following,

- we refer to a problem reduced on a given subset of features as a *restricted* problem (restricted least-squares(RLS), restricted LASSO), and to the original problem as the *full* one;
- a coefficient is qualified as *feasible* if it satisfies constraints that were ignored in its computation and tested afterwards.

Definition 5. *A necessary active set is one on which the restricted LASSO zeroes none of the coefficients, i.e. presents no selection property.*

Definition 6. *A sufficient active set is one on which the restricted LASSO achieves an optimal loss (the same as the full LASSO), i.e. solving the restricted LASSO yields a full LASSO solution.*

Property 1. *An active set is optimal if and only if it is necessary and sufficient.*

Proof. This property is a trivial rewriting of the definition of a LASSO solution, given the following property that may be worth noting: given a LASSO problem on a set \mathcal{D} of features, and a solution $s = (\sigma, \boldsymbol{\beta})$, any feature that is not in σ can be removed from \mathcal{D} without affecting s being a solution of the new problem. \square

Property 2. *On a necessary active set σ , the solution of the restricted linearly constrained least-squares*

$$\underset{\boldsymbol{\beta} \in \mathbb{R}^{|\sigma|}}{\text{minimize}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \boldsymbol{\beta}\|_2^2 \quad \text{s.t.} \quad \mathbf{w}^\top \boldsymbol{\beta} \leq t$$

(in which the positivity constraint is dropped) does satisfy the positivity constraint

$$\boldsymbol{\beta} \geq \mathbf{0}$$

(it is feasible).

Proof. If the minimizer of the loss on the linear constraint plane lies outside the positivity simplex (the positivity, that was not imposed, is not verified), and since the loss is convex, the minimizer on this simplex (with positivity constraints, i.e. the restricted LASSO solution) lies on its boundary, thus some components have a zero value, and the active is not necessary. \square

It should be noted that, although – by definition – a sufficient active set contains (is a superset of) an optimal active set, a necessary active set need not be a subset of an optimal one. The necessity of an active set only means that all of its features are necessary (that is selected) in the restricted LASSO solution.

2.1.2 The computation step

Let us then assume the optimality of a given active set σ . This allows to discard the inactive features and just consider the corresponding feature matrix and penalization vector, that we shall just note \mathbf{X} and \mathbf{w} . This also allows, by property 2, to discard the positivity constraint, leaving only the oblique one. Positivity can be tested afterwards to assess the assumed sufficiency of the active set. Hence we reduce (1.17) to

$$\text{minimize } \|\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}\|^2 \quad (2.1)$$

$$\text{subject to } \mathbf{w}^\top \boldsymbol{\beta} \leq t \quad (2.2)$$

If the constraint (2.2) is active, that is if the unconstrained solution of (2.1) violates it, then, by simple convexity arguments, the optimizer lies on the boundary of the constraint, and the problem is equivalent to

$$\text{minimize } \|\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}\|^2 \quad (2.3)$$

$$\text{subject to } \mathbf{w}^\top \boldsymbol{\beta} = t \quad (2.4)$$

Let us solve this simplified problem, regardless of the actual activity of the constraint, since its possible inactivity will appear in this computation.

(2.4) defines an hyperplane on which we must minimize the convex and differentiable loss function. This is a simple problem of which the unique solution is characterized by the fact that the isosurface of the loss is tangent to the hyperplane at this solution point, that is the gradient of the loss is normal to the plane. Instead of the gradient, we might consider its opposite, to have a direction that *minimizes* the loss, and might also halve it to simplify the equations. The normality of this negative half gradient to the plane can be expressed as its collinearity to the vector \mathbf{w} , which is by definition normal to the plane. Thus $\boldsymbol{\beta}$ is the unique solution of (2.3)(2.4) if and only if there exists a multiplier $\lambda \in \mathbb{R}$ such that

$$-\frac{1}{2} \nabla_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}\|_2^2 = \lambda \mathbf{w} \quad (2.5)$$

where the value of the constant factor to the gradient does not affect the validity of the statement, and a value of $-\frac{1}{2}$ simplifies further expressions:

$$(2.5) \iff \mathbf{X}(\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}) = \lambda \mathbf{w}$$

$$(2.5) \iff \boldsymbol{\beta} = (\mathbf{X}\mathbf{X}^\top)^{-1}(\mathbf{X}\mathbf{y} - \lambda \mathbf{w})$$

which can be noted

$$(2.5) \iff \boldsymbol{\beta} = \underbrace{(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{y}}_{\boldsymbol{\beta}^*} - \lambda \underbrace{(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{w}}_{\boldsymbol{\Delta}\boldsymbol{\beta}} \quad (2.6)$$

where $\boldsymbol{\beta}^*$ is the unconstrained solution of (2.3), and $\boldsymbol{\Delta}\boldsymbol{\beta}$ can be seen as a regularization direction.

The value of λ can be obtained by substituting $\boldsymbol{\beta}$ in (2.4):

$$\begin{aligned} \mathbf{w}^\top(\boldsymbol{\beta}^* - \lambda\boldsymbol{\Delta}\boldsymbol{\beta}) &= t \\ \lambda &= \frac{\mathbf{w}^\top\boldsymbol{\beta}^* - t}{\mathbf{w}^\top\boldsymbol{\Delta}\boldsymbol{\beta}} \end{aligned} \quad (2.7)$$

(2.6) and (2.7) transform – for a fixed, known, active set – the constraint parameter t into a regularization parameter λ , that quantifies how much the constraint takes the solution away from the unconstrained one $\boldsymbol{\beta}^*$, along a linear direction. λ is a decreasing function of t , as appears trivially in a geometrical representation (a larger feasible region gets closer to the exterior point $\boldsymbol{\beta}^*$), and appears analytically in (2.7) from the fact that the denominator $\mathbf{w}^\top\boldsymbol{\Delta}\boldsymbol{\beta}$ is equal to $\mathbf{w}^\top(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{w}$ and $(\mathbf{X}\mathbf{X}^\top)^{-1}$ being positive-definite, this denominator is positive, whence the factor of t is negative.

Since the computation that was just described involves more or less explicitly that of the unconstrained solution $\boldsymbol{\beta}^*$, the assumption that the constraint is active can be checked by verifying that its ℓ^1 -norm $\mathbf{w}^\top\boldsymbol{\beta}^*$ is greater than t . This is equivalent to λ being positive. If it is negative, the constraint is inactive, and the solution is the restricted least-squares $\boldsymbol{\beta}^*$ corresponding to $\lambda = 0$.

This gives the following general solution of the LASSO, given an optimal active set σ :

$$\begin{cases} \boldsymbol{\beta} &= \boldsymbol{\beta}^* - \lambda\boldsymbol{\Delta}\boldsymbol{\beta} \\ \boldsymbol{\beta}^* &= (\mathbf{X}_\sigma\mathbf{X}_\sigma^\top)^{-1}\mathbf{X}_\sigma\mathbf{y} \\ \boldsymbol{\Delta}\boldsymbol{\beta} &= (\mathbf{X}_\sigma\mathbf{X}_\sigma^\top)^{-1}\mathbf{w}_\sigma \\ \lambda &= \max(0, \frac{\mathbf{w}_\sigma^\top\boldsymbol{\beta}^* - t}{\mathbf{w}_\sigma^\top\boldsymbol{\Delta}\boldsymbol{\beta}}) \end{cases}$$

Figure 2.1 gives a geometrical illustration of this computation of a solution on an optimal active set. Figure 2.2 illustrates the case where the assumed feasibility (positivity of the coefficients) is not fulfilled, thus indicating that the optimizer lies on a reduced active set.

2.2 The active set descent method

The simplex algorithm ((Dantzig et al., 1954)) for solving linear programs, despite a worst-case exponential complexity, has proven remarkably efficient in practice,

and easy to implement. The same principles have been applied to quadratic programming with similar results, in different settings and formulations. These algorithms are usually referred to as the family of *active set* methods. Their main common principle is to solve a minimization problem subject to constraints by repeatedly solving sub-problems defined by a set of *active* constraints and updating this set. An exposition of the most general forms of these algorithms can be found in (Nocedal and Wright, 2000).

It should be first clarified that the term *active set* refers here to the more general concept of *constraints activity*, as opposed to variable/feature activity. However, these concepts usually coincide in practice, as will be seen below.

Regarding the plural form in *methods*, it usually refers, as in (Nocedal and Wright, 2000), to the derivation of the same method for different settings, like convex or indefinite QP, or for different formulations (primal, dual, primal–dual). It is also legitimate to consider the two other algorithms studied here (homotopy and cyclical coordinate descent) as active set methods, since they do update a set of active features/constraints; there is, at least, a possibility of confusion about which methods may be included in this category.

We focus here on the general primal active set method for convex programming. We start by giving its most simple and general form, that is for convex but not necessarily quadratic programming. Surprisingly, no such exposition was found in existing literature, and although it may not be of direct practical use, it gives the core of the algorithm, which allows to understand and derive it more easily. We then specify it for the case of convex quadratic programs without covering all aspects in details, and present its application to the more specific and simple case of LASSO. In order to clarify the taxonomy of algorithms, we refer to this *primal* method for *convex* programming, as the *Active Set Descent* (ASD) method.

2.2.1 The active set descent method for convex programming

The general idea of the method is to follow a path on the surface of the constraint polytope that corresponds to a sequence of adjacent active sets (a constraint is activated or inactivated at each step), and on which the objective function is monotonically decreasing to its minimum. The changes in the active set are determined by considerations of sufficiency and necessity of the active set.

Let us first define the problem using, for clarity, the classical notations of optimization, in which \mathbf{x} , or its components x_i , are the variables on which a function is optimized; they will correspond to the coefficient β when applying to regression, and should not be confused with the predictors. Let f be a strictly convex function over \mathbb{R}^m . We consider the problem of minimizing f under linear inequality and equality constraints over its argument:

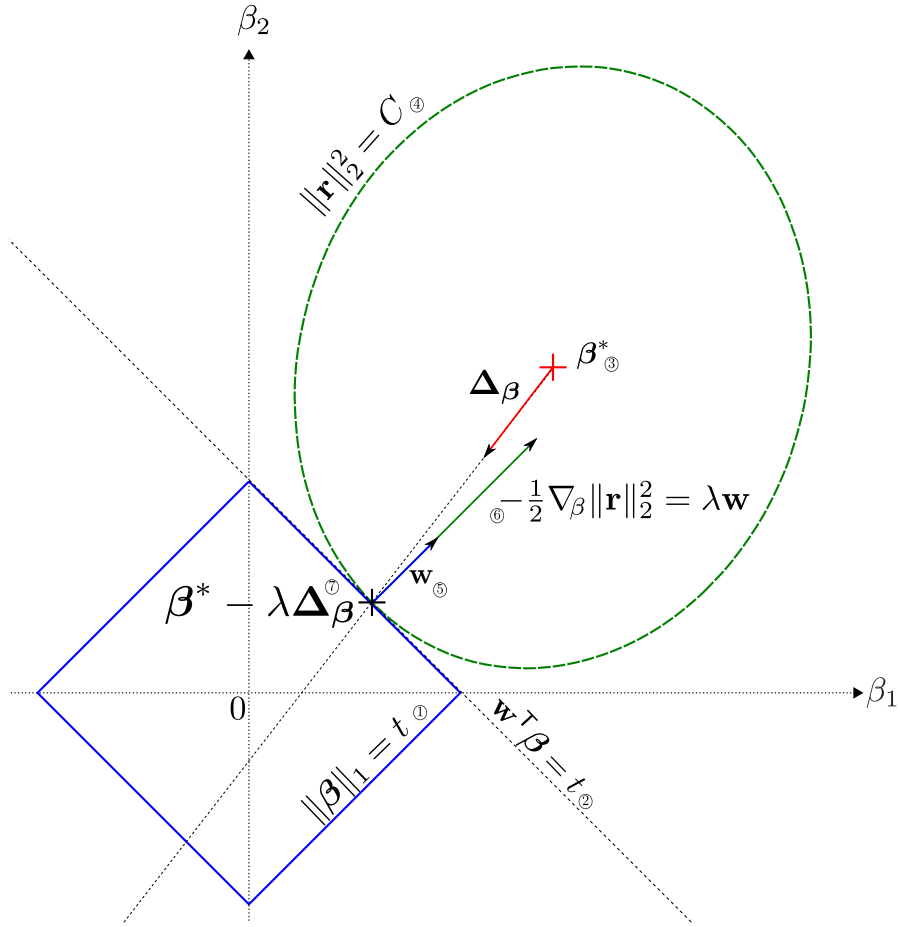


Figure 2.1: Geometrical representation of the computation of a solution on a known, or assumed, optimal active set $\{+\phi_1, +\phi_2\}$.

\mathbf{r} is the residual $\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}$.

- ① the constraint polytope,
- ② $\mathbf{w}^\top \boldsymbol{\beta}$ is its linearization, β_1 and β_2 being assumed/known to be positive (eq. (2.4)),
- ③ $\boldsymbol{\beta}^*$ is the least-squares solution (eq. (2.3)) with features ϕ_1 and ϕ_2 ,
- ④ an isocurve of the squared residual, C being the smallest value for which it reaches the constraint polytope,
- ⑤ \mathbf{w} is normal to the constraint
- ⑥ $-\frac{1}{2} \nabla_{\boldsymbol{\beta}} \|\mathbf{r}\|_2^2$ is normal to the isocurve, hence collinear with \mathbf{w} (eq. (2.5)),
- ⑦ deriving (2.5)) gives the solution of (2.3)(2.4) as a translation of $\boldsymbol{\beta}^*$ along a regularization direction $\Delta \boldsymbol{\beta}$.

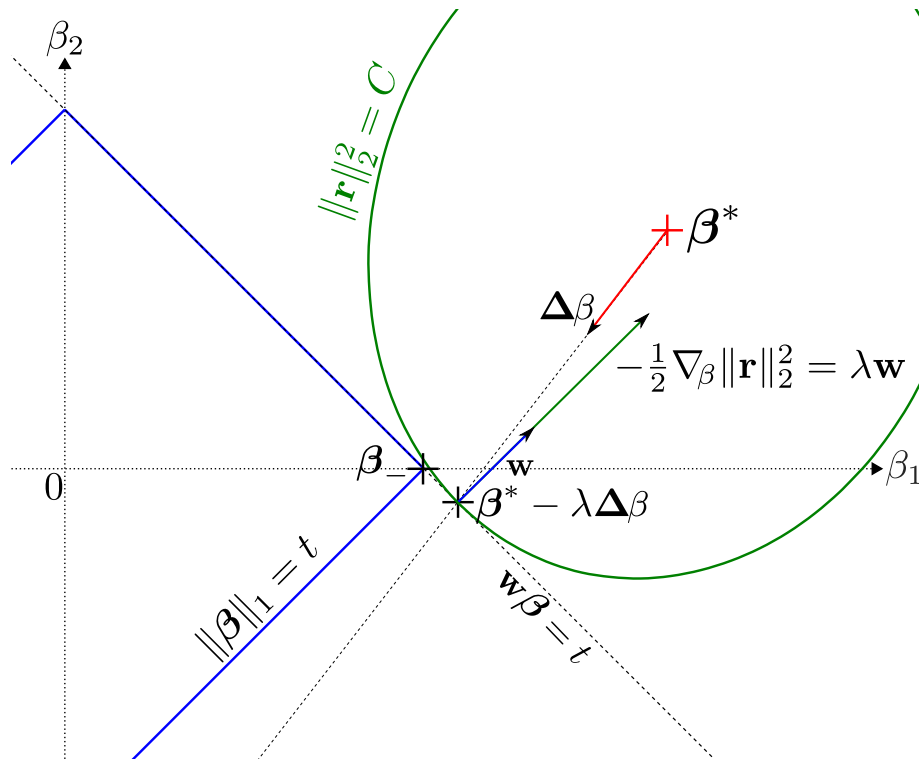


Figure 2.2: From an assumed optimal active set $\{+\phi_1, +\phi_2\}$, disagreement of the regularized least-square with the sign hypothesis, indicating a better solution β_- on the reduced active set $\{+\phi_1\}$.

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad (2.8)$$

$$\text{s.t.} \begin{cases} \mathbf{g}_i^\top \mathbf{x} \geq a_i, & i = 1, \dots, n_g \\ \mathbf{h}_i^\top \mathbf{x} = b_i, & i = 1, \dots, n_h \\ \mathbf{l}_i^\top \mathbf{x} \leq c_i, & i = 1, \dots, n_l \end{cases} \quad (2.9)$$

By adding and changing variables, this problem can be reformulated in an extended canonical form that simplifies solving algorithms, in which inequality constraints become positivity constraints. However, this reformulation will not be necessary in the problems studied in this chapter, therefore we do not detail this transformation.

Let us formalize the generalized definition of an active set by the following:

An inequality constraint is qualified as *violated* if it is not satisfied, *active* if equality holds, and *inactive* when a strict inequality holds. Similarly, an equality constraint is either violated or active. The active set associated to a point \mathbf{x} with respect to a set of constraints can be defined as the set of all active constraints, or that of all inactive constraints. These two definitions are equally worth in the following, because only feasible points will be considered (no constraint is violated), and an active set serves as an indication of which ones are active and which ones are not. We do not actually mention active sets in the remaining of this subsection, but simply the possibility of distinguishing between active and inactive constraints.

The algorithm consists in making a variable \mathbf{x} form a descent path by the following rules:

1. all constraints that are active (equality holds) are temporarily *fixed* as active and become forced equalities;
2. all inactive constraints are disregarded to compute a minimizer only subject to the active (equality) constraints, toward which \mathbf{x} is moved up to the first point where a constraint is activated, if any;
3. when an optimal and feasible point w.r.t. these forced constraints has been reached, improve, if possible, by inactivating one of them.

In rule 3, the constraint is more precisely *considered* as inactive, or made *potentially* inactive: although no change is made in \mathbf{x} – leaving the constraint active – the equality is not fixed anymore, which leads to its actual inactivation by rule 1. It should also be noted that equality constraints cannot be inactivated, since they can only be active or violated. The formal sketch of the method is given in Algorithm 1.

The convergence of this algorithm is ensured by the descent property: the objective function is decreasing at each inactivation or activation of a constraint, and stopping only at a local minimum which is also global from the convexity of f . As for the simplex algorithm, a cycle can occur and prevent convergence only

Algorithm 1 active set descent method for convex programming

Inactivation consists in allowing an *inequality* constraint to hold strictly (inactive) rather than being forced into an equality (active constraints) in line 4

Input: convex function f , linear equality and inequality constraints

Output: a minimizer of f subject to the constraints

```

1:  $\mathbf{x} \leftarrow$  feasible point:
    all constraints are satisfied, and either inactive (strict inequality) or ac-
    tive (equality)
2: loop
3:   repeat
4:     find a minimizer  $\bar{\mathbf{x}}$  of  $f$  subject to the active constraints
5:     move  $\mathbf{x}$  linearly towards  $\bar{\mathbf{x}}$  ( $\mathbf{x} \leftarrow \mathbf{x} + \gamma(\bar{\mathbf{x}} - \mathbf{x})$ ,  $\gamma$  going from 0 to 1)
        up to the first activation of an inactive constraint.
6:   until no new activation
7:   if inactivating an active constraint yields an improvement then
8:     consider it inactive
9:   else
10:    return  $\mathbf{x}$ 
11:  end if
12: end loop

```

in some degenerate cases where a constraint inactivation yields an immediate activation of at least two other constraints, leading to successive activations and inactivations without decreasing in f , possibly looping. Most of the analyses and workarounds made on the simplex algorithm, (see (Dantzig, 1998) for a review), like the lexicographic method, can be transposed to the active set method. If no such unproductive inactivation occurs, there can be no cycle, since this can only happen on a sequence where f is not decreasing, meaning a sequence containing only constraint activations, and a constraint cannot be activated twice without being inactivated in between.

The number of steps is guaranteed to be less than the number of possible active sets, since each time one is considered, f is minimized over it, hence all subsequent steps, where f is necessarily lower, necessarily involves different active sets.

2.2.1.1 Case of a differentiable objective function

A practical implementation of the method is made easier by the differentiability of the objective function f and by the reformulation of the problem in a canonical form.

The differentiability of f , or more precisely its sub-differentiability, allows to determine if inactivating a constraint can decrease f , and is of great help for the step of minimizing it under the active constraints.

The reformulation consists in associating an additional *slack* variable to each inequality constraint, so that they are equivalent to their associated variable being positive, and then rewrite \mathbf{x} as a (linear) function of these new variables, so that each variable is associated to a (positivity) constraint. Up to some maintenance of the rewriting of \mathbf{x} , the problem is then equivalent to one of *canonical* form

$$\begin{aligned} & \text{minimize } f(\mathbf{x}) \\ & \text{subject to } \begin{cases} \mathbf{Ax} = \mathbf{b} \\ \mathbf{x} \geq 0 \end{cases} \end{aligned}$$

in which inactivating a constraint is equivalent to activating a variable. This form allows to define the active set, as previously, as the set of active *variables*.

The operation of minimizing f under the active constraints (i.e. over the active variables) can be handled in different ways for the native equalities $\mathbf{Ax} = \mathbf{b}$ (Lagrange multipliers or variable elimination), and by direct elimination for the activated constraints, that amounts to zeroing a variable. In all cases, the constrained minimization is equivalent to the simple minimization of an alternative or reformulated function (the Lagrangian or the rewriting with eliminated variables) that we refer to as \bar{f}_σ .

In the step of taking \mathbf{x} to $\bar{\mathbf{x}}$ up to the first activation of an inactive constraint, one must now monitor which active variable becomes inactive (is zeroed) first, on the way from \mathbf{x} to $\bar{\mathbf{x}}$. x_i being an active variable ($i \in \sigma$), let γ_i be the translation factor that zeroes x_i :

$$x_i + \gamma_i(\bar{x}_i - x_i) = 0$$

i.e.

$$\gamma_i = \frac{\bar{x}_i - x_i}{x_i}$$

The first zeroing on the way from \mathbf{x} to $\bar{\mathbf{x}}$ translates to the lowest among those that lie in $[0, 1]$:

$$\gamma = \arg \min_{i \in \sigma} \left(\frac{\bar{x}_i - x_i}{x_i} \right) \in [0, 1]$$

The step where constraints are considered for inactivation now consists in finding an inactive variable x_i with respect to which the derivative of $\bar{f}_{\sigma \cup \{i\}}$ is negative (thus lessening \bar{f} when getting a positive coefficient). As for the simplex algorithm, the greedy strategy of activating the one having the most negative derivative is generally a good choice.

The resulting specification of algorithm 1 for differentiable canonical convex programs is given in algorithm 2.

Algorithm 2 active set descent for differentiable canonical convex programming

Definition: For any set σ of inactive constraints (active set), let \bar{f}_σ be a function such that $\arg \min \bar{f}_\sigma = \arg \min f$ s.t. all active constraints (initial ones and those induced by σ), using Lagrange multipliers or variable elimination.

Input: differentiable convex function f , linear equality constraints, and non-negativity constraints on certain variables

Output: a minimizer of f subject to the constraints

```

1:  $\mathbf{x} \leftarrow$  feasible point (usually  $\mathbf{0}$  or an informed guess)
2:  $\sigma \leftarrow \{i \mid x_i > 0\}$ 
3: loop
4:   repeat
5:      $\bar{\mathbf{x}} \leftarrow \arg \min_{\mathbf{a}} \bar{f}_\sigma(\mathbf{a})$ 
6:      $(\gamma, i) \leftarrow \min, \arg \min_{i \in \sigma} \left( \frac{x_i}{x_i - \bar{x}_i} \right)_{\geq 0}$ 
7:     if  $\gamma \leq 1$  then
8:        $\mathbf{x} \leftarrow \mathbf{x} + \gamma(\bar{\mathbf{x}} - \mathbf{x})$ 
9:        $\sigma \leftarrow \sigma \setminus \{i\}$ 
10:    end if
11:  until  $\gamma \geq 1$ 
12:   $\mathbf{x} \leftarrow \bar{\mathbf{x}}$ 
13:   $i \leftarrow \arg \min_{i \notin \sigma} \frac{\partial \bar{f}_{\sigma+i}(\mathbf{x})}{\partial x_i}$ 
14:  if  $\frac{\partial \bar{f}_{\sigma+i}(\mathbf{x})}{\partial x_i} < 0$  then
15:     $\sigma \leftarrow \sigma \cup \{i\}$ 
16:  else
17:    return  $\mathbf{x}$ 
18:  end if
19: end loop

```

2.2.2 active set descent for the LASSO

Let us first reproduce in Algorithm 3 the sketch of the NNLS algorithm – mentioned in section 1.2.1 – as published in (Lawson and Hanson, 1974), so that the reader can verify that it matches the active set descent applied to least-squares with non-negativity constraints. Thus the task of deriving ASD for the LASSO problem is equivalent to that in (Tibshirani, 1996) of deriving NNLS for the same problem, and to the derivation of the active set method in (Osborne et al., 2000a), although ours and (Osborne et al., 2000a) give a different result than (Tibshirani, 1996), and ours gives a different insight of the method, and different further adaptations, compared to (Osborne et al., 2000a).

Algorithm 3 NNLS algorithm as given in (Lawson and Hanson, 1974)

(23.10) ALGORITHM NNLS($E, m_2, n, f, x, w, z, \mathcal{O}, \mathbf{Z}$)

<i>Step</i>	<i>Description</i>
1	Set $\mathcal{O} := \text{NULL}$, $\mathbf{Z} := \{1, 2, \dots, n\}$, and $x := 0$.
2	Compute the n -vector $w := E^T(f - Ex)$.
3	If the set \mathbf{Z} is empty or if $w_j \leq 0$ for all $j \in \mathbf{Z}$, go to Step 12.
4	Find an index $t \in \mathbf{Z}$ such that $w_t = \max\{w_j : j \in \mathbf{Z}\}$.
5	Move the index t from set \mathbf{Z} to set \mathcal{O} .
6	Let $E_{\mathcal{O}}$ denote the $m_2 \times n$ matrix defined by
	$\text{Column } j \text{ of } E_{\mathcal{O}} := \begin{cases} \text{column } j \text{ of } E & \text{if } j \in \mathcal{O} \\ 0 & \text{if } j \in \mathbf{Z} \end{cases}$
	Compute the n -vector z as a solution of the least squares problem $E_{\mathcal{O}}z \cong f$. Note that only the components $z_j, j \in \mathcal{O}$, are determined by this problem. Define $z_j := 0$ for $j \in \mathbf{Z}$.
7	If $z_j > 0$ for all $j \in \mathcal{O}$, set $x := z$ and go to Step 2.
8	Find an index $q \in \mathcal{O}$ such that $x_q/(x_q - z_q) = \min\{x_j/(x_j - z_j) : z_j \leq 0, j \in \mathcal{O}\}$.
9	Set $\alpha := x_q/(x_q - z_q)$.
10	Set $x := x + \alpha(z - x)$.
11	Move from set \mathcal{O} to set \mathbf{Z} all indices $j \in \mathcal{O}$ for which $x_j = 0$. Go to Step 6.
12	Comment: The computation is completed.

As shown in the previous chapter, the LASSO program, when transformed by the positivity trick, is indeed a convex quadratic program in canonical form, except for the single oblique inequality, and with the property that a variable

and its opposite are mutually exclusive and can be handled by pair.

Subsection 2.1.2 gives the minimization step that can be plugged in active set descent. The technique of assuming that the LASSO is active ($\beta_{\text{least-squares}} \neq \beta_{\text{LASSO}}$) to cast the inequalities $\|\beta\|_{\mathbf{w}} \leq t$ or $\mathbf{w}^T \beta \leq t$ into equalities, which was used at this minimization level, can alternatively be used at the main-loop level of the algorithm: activity/equality is assumed from the start, and this assumption can be checked after convergence from the sign of the last Lagrange multiplier, its negativity showing that $\beta_{\text{LASSO}} = \beta_{\text{least-squares}}$.

2.2.2.1 Plugging the LASSO program in the active set descent method

Let us switch back to the regression notations. To ease the transitions, let us first assume a finite set of features $\mathcal{D} = \{\phi_1, \dots, \phi_m\}$, extended by their opposites $\{-\phi_1, \dots, -\phi_m\}$, define an active set by a list of integer indices $\sigma \subset \{1, \dots, m\} \cup \{-1, \dots, -m\}$ and note $\phi_{-i} = -\phi_i$. Let us note \mathbf{X} the full predictor matrix, and \mathbf{X}_σ its restrictions to the active set, and use the same subscript notation for β and \mathbf{w} . For example,

$$\mathbf{X}_{(4,-2)} = \begin{pmatrix} - & \phi_4 & - \\ - & -\phi_2 & - \end{pmatrix}.$$

The penalization $w(\phi)$ is always positive, thus it remains the same for $+\phi$ and $-\phi$. Therefore, the restriction of \mathbf{w} to the active set ignores the signs in σ :

$$\mathbf{w}_{(4,-2)} = \begin{pmatrix} w_4 \\ w_2 \end{pmatrix}$$

Concerning β , its restriction to σ is the object the algorithm is building, and there is an extension of β_σ to β – when returning a solution, rather than a restriction: given a sparse solution, e.g. $\beta_\sigma = (9, 8)^T$ with $\sigma = (4, -2)$, the complete solution vector is $\beta = (0, -8, 0, 9, \dots, 0, \dots)^T$.

The function to be minimized in the constrained formulation of the LASSO is

$$f(\beta) = \|\mathbf{y} - \mathbf{X}^T \beta\|_2^2.$$

The same function constrained to an active set σ is

$$f_\sigma(\beta_\sigma) = \|\mathbf{y} - \mathbf{X}_\sigma^T \beta_\sigma\|_2^2.$$

We have seen in section 2.1.2 that the minimizer of the latter subject to the additional constraint $\mathbf{w}_\sigma^T \beta_\sigma \leq t$ is characterized by the tangency equation (2.5); this implies that it is also the unique minimizer of the corresponding Lagrangian

Λ :

$$\begin{aligned}
(2.5) : \quad & -\frac{1}{2}\nabla_{\beta_\sigma} f_\sigma(\beta_\sigma) = \lambda \mathbf{w}_\sigma \\
& \iff \frac{1}{2}\nabla_{\beta_\sigma} f_\sigma(\beta_\sigma) + \lambda \mathbf{w}_\sigma = \mathbf{0} \\
& \iff \nabla_{\beta_\sigma} \left[\frac{1}{2}f_\sigma(\beta_\sigma) + \lambda \mathbf{w}_\sigma^\top \beta_\sigma \right] = \mathbf{0} \\
& \iff \beta_\sigma \in \arg \min_{\beta} \underbrace{\left[\frac{1}{2}f_\sigma(\beta) + \lambda \mathbf{w}_\sigma^\top \beta \right]}_{\Lambda(\beta)}
\end{aligned}$$

This gives us the function needed for the algorithm: $\bar{f}_\sigma = \Lambda$, of which the gradient is

$$\nabla_{\beta_\sigma} \bar{f}_\sigma(\beta_\sigma) = -\mathbf{X}_\sigma(\mathbf{y} - \mathbf{X}_\sigma^\top \beta_\sigma) + \lambda \mathbf{w}_\sigma.$$

Its zeroing is equivalent to the *correlation* of all features being equal to λ times their penalization:

$$-\mathbf{X}_\sigma(\mathbf{y} - \mathbf{X}_\sigma^\top \beta_\sigma) + \lambda \mathbf{w}_\sigma = 0 \iff \forall \phi \in \sigma, \phi^\top (\mathbf{y} - \mathbf{X}_\sigma^\top \beta_\sigma) = \lambda w(\phi)$$

but a complication may seem to arise in the fact that the Lagrange multiplier is dependent on the active set: when testing if activating a feature yields a better solution, the function to minimize should be that associated to the augmented set $\frac{\partial \bar{f}_{\sigma+i}}{\partial \beta_i} = \frac{1}{2} \frac{\partial f}{\partial \beta_i} + \lambda_{\sigma+i} w_i$. Unfortunately, the Lagrange multiplier $\lambda_{\sigma+i}$ is not known, and although its computation is possible, it amounts to solving a least-squares problem, which is not desirable. Fortunately, the following property holds:

Property 3. *In order to test if the activation of a feature ϕ_i , a surrogate function $\tilde{f}_{\sigma+i}(\beta_{\sigma+i}) = \frac{1}{2} \|\mathbf{y} - \mathbf{X}_{\sigma+i}^\top \beta_{\sigma+i}\|_2^2 + \lambda_\sigma \mathbf{w}_{\sigma+i}^\top \beta_{\sigma+i}$, that just -potentially- activates ϕ_i , but keeps the current Lagrange multiplier, can be used.*

Proof. This is a direct application of the Kuhn-Tucker conditions (introduced in (Kuhn and Tucker, 1951), see e.g. (Nocedal and Wright, 2000) or (Nash and Sofer, 1996)), that are both sufficient and necessary for a convex objective function. However, to follow the purpose of giving in this thesis a “standalone” understanding of the concepts, the proof is given for this special case involving a single Lagrange multiplier.

When no improvement is obtained from this augmented set, $\tilde{f}_{\sigma+i}$, the minimizer on both sets is the same, which is characterized by the correlation of features from σ being equal to λ_σ . Therefore, \bar{f}_σ is a valid function for minimizing over $\sigma + i$, and its gradient with respect to β_i is positive. This means that, by opposition, a feature can be activated on the basis of the associated gradient being non-positive:

$$\frac{\partial \tilde{f}_{\sigma+i}}{\partial \beta_i} \leq 0 \Rightarrow \arg \min_{\beta} f_{\sigma+i}(\beta) \text{ s.t. } \mathbf{w}^\top \beta \leq \arg \min_{\beta} f_\sigma(\beta) \text{ s.t. } \mathbf{w}_\sigma^\top \beta \quad (2.10)$$

We then show that a positive gradient on $\tilde{f}_{\sigma+i}$ implies that the restricted LASSO minimizer on σ is also the restricted LASSO minimizer on $\sigma+i$. Let β_1 be this minimizer. If, when extending the active set with feature ϕ_i , the gradient of the extended function $\tilde{f}_{\sigma+i}$ wrt. β_i is positive, β_1 remains the minimizer, noting β_2 the LASSO solution on $\sigma+i$, it satisfies

$$\|\mathbf{y} - \mathbf{X}_{\sigma+i}^T \beta_2\|_2^2 + \lambda_\sigma \mathbf{w}_{\sigma+i}^T \beta_2 \geq \|\mathbf{y} - \mathbf{X}_{\sigma+i}^T \beta_1\|_2^2 + \lambda_\sigma \mathbf{w}_{\sigma+i}^T \beta_1$$

Being a LASSO solution, it satisfies the positivity constraint:

$$\mathbf{w}_{\sigma+i}^T \beta_2 = \|\beta_2\|_{\mathbf{w}},$$

hence

$$\|\mathbf{y} - \mathbf{X}_{\sigma+i}^T \beta_2\|_2^2 - \|\mathbf{y} - \mathbf{X}_{\sigma+i}^T \beta_1\|_2^2 \geq +\lambda_\sigma (\|\beta_1\|_{\mathbf{w}} - \|\beta_2\|_{\mathbf{w}}).$$

We have assumed the LASSO constraint $\|\beta\|_{\mathbf{w}} \leq t$ to be active, that is, for the LASSO solution, and for all restricted LASSO solutions, $\|\beta_1\|_{\mathbf{w}} = t$, the second term is thus zero, whence,

$$\|\mathbf{y} - \mathbf{X}_{\sigma+i}^T \beta_2\|_2^2 = \|\mathbf{y} - \mathbf{X}_{\sigma+i}^T \beta_1\|_2^2$$

and β_1 is a restricted LASSO solution on $\sigma+i$. Thus,

$$\frac{\partial \tilde{f}_{\sigma+i}}{\partial \beta_i} > 0 \Rightarrow \arg \min_{\beta} f_{\sigma+i}(\beta) \text{ s.t. } \mathbf{w}^T \beta = \arg \min_{\beta} f_{\sigma}(\beta) \text{ s.t. } \mathbf{w}_{\sigma}^T \beta \quad (2.11)$$

□

This specification of Algorithm 2 to the LASSO problems is formalized in Algorithm 4.

In line 16 of the algorithm, where inactive features are considered for activation, the minimum is taken over the doubled set that includes the negative features. This is the only point where the doubling may induce a computational cost. However, a feature ϕ_i and $-\phi_i$ can be considered at the same time, as the sign of $\phi_i(\mathbf{y} - \mathbf{X}_{\sigma}^T \beta_{\sigma})$ determines which of them gives the lowest value: following the simple equality

$$\min(a - b, a - (-b)) = a - |b|,$$

line 16 can be replaced by

$$i = \arg \min_{i \in \{1, \dots, m\} \setminus \sigma} [g(i) = \lambda_i w_i - |\phi_i(\mathbf{y} - \mathbf{X}_{\sigma}^T \beta_{\sigma})|]$$

and the sign of the selected feature is that of $\phi_i(\mathbf{y} - \mathbf{X}_{\sigma}^T \beta_{\sigma})$. This makes the doubling of the feature set more virtual, as it now only appears through the use of negative signs in σ , which can equally be seen as discriminating ϕ_i 's and

Algorithm 4 active set descent for the linear LASSO (predictors $\in \mathbb{R}^p$)

This is a direct application of Algorithm 2 after turning the LASSO into a QP problem by the positivity trick.

Input: response vector $\mathbf{y} \in \mathbb{R}^n$, predictor matrix $\mathbf{X} = \begin{pmatrix} - & \phi_1 & - \\ & \vdots & \\ - & \phi_p & - \end{pmatrix} \in \mathbb{R}^{n \times p}$, penalization vector $\mathbf{w} \in \mathbb{R}_+^p$, constraint $t \in \mathbb{R}_+$

Output: $\text{LASSO}_c(\mathbf{X}, \mathbf{y}, \mathbf{w}, t) = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \|\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}\|_2^2$ s.t. $\|\boldsymbol{\beta}\|_{\mathbf{w}} \leq t$

Definition: $\mathcal{I} = \{-p, \dots, -1, 1, \dots, p\}$, $\phi_{-i} = -\phi_i$

- 1: $\boldsymbol{\beta} \leftarrow$ feasible point ($\|\boldsymbol{\beta}\|_{\mathbf{w}} \leq t$, usually $\mathbf{0}$ or an informed guess)
- 2: $\sigma \leftarrow \{\text{sign}(\beta_i)i \mid \beta_i \neq 0\}$
- 3: **loop**
- 4: **repeat**
- 5: $\boldsymbol{\beta}^* \leftarrow (\mathbf{X}_\sigma \mathbf{X}_\sigma^\top)^{-1} \mathbf{X}_\sigma \mathbf{y}$
- 6: $\boldsymbol{\Delta} \boldsymbol{\beta} \leftarrow (\mathbf{X}_\sigma \mathbf{X}_\sigma^\top)^{-1} \mathbf{w}_\sigma$
- 7: $\lambda \leftarrow \frac{\mathbf{w}^\top \boldsymbol{\beta}^* - t}{\mathbf{w}^\top \boldsymbol{\Delta} \boldsymbol{\beta}}$
- 8: $\bar{\boldsymbol{\beta}} \leftarrow \boldsymbol{\beta}^* - \lambda \boldsymbol{\Delta} \boldsymbol{\beta}$
- 9: $(\gamma, i) \leftarrow \min, \arg \min_{i \in \sigma} \left(\frac{\beta_i}{\beta_i - \bar{\beta}_i} \right)_{\geq 0}$
- 10: **if** $\gamma \leq 1$ **then**
- 11: $\boldsymbol{\beta}_\sigma \leftarrow \boldsymbol{\beta}_\sigma + \gamma(\bar{\boldsymbol{\beta}} - \boldsymbol{\beta}_\sigma)$
- 12: $\sigma \leftarrow \sigma \setminus \{i\}$
- 13: **end if**
- 14: **until** $\gamma \geq 1$
- 15: $\boldsymbol{\beta}_\sigma \leftarrow \bar{\boldsymbol{\beta}}$
- 16: $i \leftarrow \arg \min_{i \in \mathcal{I} \setminus \sigma} [g(i) = \lambda w_i - \phi_i^\top (\mathbf{y} - \mathbf{X}_\sigma^\top \boldsymbol{\beta}_\sigma)]$
- 17: **if** $g(i) < 0$ **then**
- 18: $\sigma \leftarrow \sigma \cup \{i\}$
- 19: **else**
- 20: **return** $\boldsymbol{\beta}$
- 21: **end if**
- 22: **end loop**

$(-\phi_i)$'s, or as accounting for the sign of β_i 's. One might find it more convenient to explicitly switch back to the sign-monitoring view when defining a practical algorithm. This means handling the signs in β rather than in σ , and the emergence of the sign vector $\text{sign}(\beta)$ in some expressions. More precisely, this sign can be systematically associated to \mathbf{w}_σ , forming the gradient of the weighted ℓ^1 -norm:

$$\frac{\partial \|\beta\|_{\mathbf{w}}}{\partial \beta_i} = w_i \text{sign}(\beta_i).$$

An other noteworthy point is that the algorithm is more easily stated and implemented by using and storing explicitly the gradient that we note θ and can be thought of as a weighted sign vector, or a signed penalization vector:

$$\theta_\sigma = \nabla_{\beta_\sigma} \|\beta_\sigma\|_{\mathbf{w}_\sigma} \quad (2.12)$$

$$= (w_i \text{sign}(\beta_i))_{i \in \sigma} \quad (2.13)$$

The algorithm 5 uses this alternative formulation. It also abandons references to a finite, indexed set of features: the set \mathcal{D} of possible features is arbitrary, and these features apply to an arbitrary initial input set \mathcal{X} , σ represents a finite subset of \mathcal{D} , with no consideration on its actual representation. Given a feature ϕ , ϕ denotes the vector $(\phi(x_1), \dots, \phi(x_n))^T$ corresponding to a row of \mathbf{X} , and β_ϕ denotes the coefficient associated to ϕ when it is active.

2.2.2.2 Contribution with respect to the active set algorithm of (Osborne et al., 2000a)

Both formulations Alg. 4 and Alg. 5 are essentially the descent method presented in (Osborne et al., 2000a; Osborne et al., 2000b). In the latter, it is said to be a standard active set method of which “the interest lies in the treatment of the active constraint”. This basically refers to the way the special shape of the ℓ^1 -norm constraint is handled. We have explained the two possible approaches (positivity trick or sign monitoring) in section 1.2.3. The authors of (Osborne et al., 2000a; Osborne et al., 2000b) chose the sign-compliance method whereas we chose to use the positivity trick throughout this chapter. Another difference is that we do not include a side element of the algorithm of (Osborne et al., 2000a) which is the possible direct activation of $-\phi$ after the inactivation of a feature ϕ . This is a minor point of the algorithm that can slightly – on empirical evidences – improve the convergence time, but we wish to present, as clearly as possible, the core of this method.

Another contribution of this section is naturally to formulate the method in the larger setting of the LASSO problem that includes penalization factors and an arbitrary set of features.

But more importantly, the motivation of the present formulation and explanations is to make them as simple and natural as possible. We have taken a

Algorithm 5 active set descent for the featurized constrained LASSO (arbitrary feature set)

Sign monitoring is used, rather than positivity trick as in Algorithm 4.

Input: input vector $\mathbf{x} \in \mathcal{X}^n$, response vector $\mathbf{y} \in \mathbb{R}^n$, feature dictionary $\mathcal{D} \subset \mathbb{R}^{\mathcal{X}}$, penalization function $w : \mathcal{D} \rightarrow \mathbb{R}_+$, constraint $t \in \mathbb{R}_+$

Output: $\text{LASSO}_c(\mathbf{x}, \mathbf{y}, \mathcal{D}, \mathbf{w}, t) = \arg \min_{(\sigma, \boldsymbol{\beta}) \in 2^{\mathcal{D}} \times \mathbb{R}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \boldsymbol{\beta}\|_2^2$ s.t. $\|\boldsymbol{\beta}\|_{\mathbf{w}_\sigma} \leq t$

Definition: For a feature function $\phi \in \mathcal{D}$, the bold notation denotes the application of ϕ to the input vector \mathbf{x} : $\boldsymbol{\phi} = \boldsymbol{\phi}(\mathbf{x}) = (\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n))^\top$.

```

1:  $\sigma \leftarrow \{\}$ ,  $\mathbf{X} \leftarrow []$ ,  $\boldsymbol{\theta} \leftarrow ()$ 
2: loop
3:   repeat
4:      $\boldsymbol{\beta}^* \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{y}$ 
5:      $\boldsymbol{\Delta}\boldsymbol{\beta} \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1}\boldsymbol{\theta}$ 
6:      $\lambda \leftarrow \frac{\boldsymbol{\theta}^\top \boldsymbol{\beta}^* - t}{\boldsymbol{\theta}^\top \boldsymbol{\Delta}\boldsymbol{\beta}}$ 
7:      $\bar{\boldsymbol{\beta}} \leftarrow \boldsymbol{\beta}^* - \lambda \boldsymbol{\Delta}\boldsymbol{\beta}$ 
8:      $(\gamma, \phi) \leftarrow \min, \arg \min_{\phi \in \sigma} \left( \frac{\beta_\phi}{\beta_\phi - \bar{\beta}_\phi} \right)_{\geq 0}$ 
9:     if  $\gamma \leq 1$  then
10:        $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \gamma(\bar{\boldsymbol{\beta}} - \boldsymbol{\beta})$ 
11:        $\sigma \leftarrow \sigma \setminus \{\phi\}$ 
12:       shrink  $\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\theta}$  accordingly
13:     end if
14:   until  $\gamma \geq 1$ 
15:    $\boldsymbol{\beta} \leftarrow \bar{\boldsymbol{\beta}}$ 
16:    $\phi \leftarrow \arg \min_{\phi \in \mathcal{D}} [g(\phi) = \lambda w(\phi) - |\boldsymbol{\phi}^\top (\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta})|]$ 
17:   if  $g(\phi) < 0$  then
18:      $\sigma \leftarrow \sigma \cup \{\phi\}$ 
19:     extend  $\mathbf{X}$  by row  $\boldsymbol{\phi}$ ,  $\boldsymbol{\beta}$  by 0, and  $\boldsymbol{\theta}$  by  $w(\phi) \text{sign}(\boldsymbol{\phi}^\top (\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}))$ 
20:   else
21:     return  $(\sigma, \boldsymbol{\beta})$ 
22:   end if
23: end loop

```

somewhat different angle, trying to rely on simple notions with the least references to general results like the general theories of Lagrange multipliers, duality, and Kuhn-Tucker conditions. We also left aside technical points or issues. One of them is the possibility, and benefits, of handling the successive least-squares problems by updating a decomposition of the Gramian matrix $\mathbf{X}_\sigma \mathbf{X}_\sigma^\top$, that we discuss in section 2.4.1.

This, we hope, should give the reader a more intuitive view on the LASSO problem and the active set descent algorithm, and clarify some points like the relations between the constraint t , the multiplier λ and the active set σ . In the following, we look further into these relations and the equivalence of the constrained and regularized versions of the LASSO, before deriving the active set descent for the latter. This derivation is the main benefit of our exposition of the active set descent method, since its idea and validity arise naturally from it, whereas it did not appear to the authors of (Osborne et al., 2000a) or their readers.

2.2.2.3 On the equivalence between the constrained and regularized formulations of the LASSO

We have seen that given an optimal active set σ^* , the solution $\beta_{(t)}$ of the LASSO with constraint t is of the form

$$\beta_{(t)} = \beta_{\sigma^*}^* - \lambda_{\sigma^*} \Delta \beta_{\sigma^*}$$

where all terms depend on σ^* , but only λ depends –linearly– on t : $\lambda = \frac{\mathbf{w}^\top \beta_{\sigma^*}^* - t}{\mathbf{w}^\top \Delta \beta_{\sigma^*}}$. $\beta_{\sigma^*}^*$ is the unconstrained restricted least-squares on σ^* , $\Delta \beta_{\sigma^*}$ is a regularization direction, and λ can be seen as translating the constraint t into how much $\beta_{(t)}$ is “taken away” from $\beta_{\sigma^*}^*$ in that direction (see Fig. 2.1).

We have also seen that $\beta_{(t)}$ is the minimizer of the Lagrangian

$$\frac{1}{2} \|\mathbf{y} - \mathbf{X}_{\sigma^*}^\top \beta\|_2^2 + \lambda_{\sigma^*} \mathbf{w}_{\sigma^*}^\top \beta$$

which, given the feasibility of σ^* , is equal to

$$\frac{1}{2} \|\mathbf{y} - \mathbf{X}_{\sigma^*}^\top \beta\|_2^2 + \lambda_{\sigma^*} \|\beta\|_{\mathbf{w}_{\sigma^*}}$$

σ^* being optimal, by definition, a lower squared residual cannot be achieved –while staying within the constraint– by extending it. Hence $\beta_{(t)}$ is the minimizer of this Lagrangian over the whole set \mathcal{D} , and

$$\arg \min_{\substack{(\sigma, \beta) \in 2^{\mathcal{D}} \times \mathbb{R} \\ \|\beta\|_{\mathbf{w}_\sigma} \leq t}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \beta\|_2^2 = \arg \min_{(\sigma, \beta) \in 2^{\mathcal{D}} \times \mathbb{R}} \frac{1}{2} \|\mathbf{y} - \mathbf{X}_\sigma^\top \beta\|_2^2 + \lambda(t) \|\beta\|_{\mathbf{w}_\sigma} \quad (2.14)$$

where

$$\lambda(t) = \frac{\boldsymbol{\theta}^\top (\mathbf{X}_{\sigma^*} \mathbf{X}_{\sigma^*}^\top)^{-1} \mathbf{X}_{\sigma^*} \mathbf{y} - t}{\boldsymbol{\theta}^\top (\mathbf{X}_{\sigma^*} \mathbf{X}_{\sigma^*}^\top)^{-1} \boldsymbol{\theta}}, \quad (2.15)$$

σ^* is an optimal active set, and $\boldsymbol{\theta}$ is the penalized sign vector of the solution (see Eq. 2.12).

This establishes the equivalence of the constrained and regularized formulations of the LASSO. The inter-dependencies between the definition of the problem and parts of its solution ($\boldsymbol{\theta}$ and σ^*) seem cumbersome, but we are just interested in showing that for each value of t , there exists a positive regularization parameter λ for which the two problems admit the same solution. We are also interested in characterizing the relation between the two parameters, rather than having a computable expression of one as a function of the other.

Before examining the properties of $\lambda(t)$, let us consider how the problem itself and its solution behave as functions of t . It is trivial to note that, from the convexity of the objective, the solution evolves continuously with t .

Another trivial result is that if \mathcal{D} is finite, so is the number of possible active sets, which implies that optimal active set remain optimal on a whole nonempty interval of the parameter.

The limit values of t are

- $t_0 = 0$ for which the only admissible solution is $\sigma = \emptyset$ (i.e. $\boldsymbol{\beta} = \mathbf{0}$),
- t_{LS} for which the constraint first reaches an unconstrained least-squares solution.

Thus the problem path for t going between these values yields a continuous path of solutions, defined on successive active sets. On an interval for which the optimal active set is constant, the expression (2.15) for λ indicates that λ evolves linearly with t . Moreover, it is decreasing in t , as previously mentioned, from the positive-definiteness of $(\mathbf{X}\mathbf{X}^\top)^{-1}$.

The continuity of the problems and solutions of the constrained form and the similar continuity of the regularized form induce that λ is necessarily continuous in t . Thus λ is a continuous, decreasing, piecewise linear function of t , of which an example can be seen in figure 2.3.

The limit values of λ are

- $\lambda_{\text{LS}} = 0$ for which the problem is an unconstrained least-squares,
- λ_0 beyond which the constraint dominates the objective and the solution is $\sigma = \emptyset$.

Contrary to the bound t_{LS} that cannot be computed independently of the corresponding solution, λ_0 corresponds to the known, trivial solution $\sigma = \emptyset$, and can be computed *a priori* from the following.

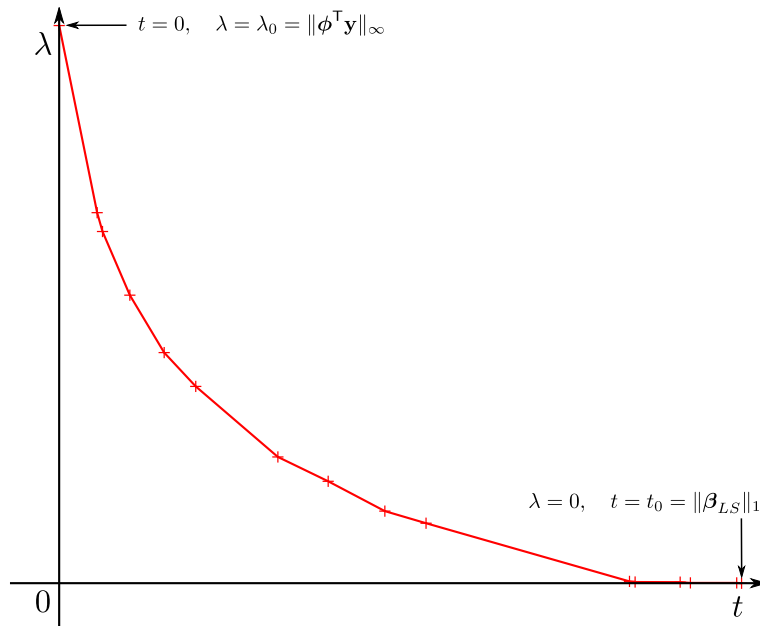


Figure 2.3: An example of the continuous, decreasing, and piecewise linear relation between the parameters λ and t of equivalent constrained and regularized forms of a LASSO problem.

\emptyset is an optimal active set if and only if no feature activation can decrease the objective, which in terms of derivatives gives

$$\forall \phi \in \mathcal{D}, \quad \phi^\top (\mathbf{y} - \mathbf{0}) \pm \lambda w_\phi \geq 0,$$

and the smallest value of λ such that this is verified is

$$\lambda_0 = \max_{\phi \in \mathcal{D}} \frac{|\phi^\top \mathbf{y}|}{w_\phi}$$

Another interesting point to note about the path of parameters in between their limit values is about the break points where the active set changes. From the continuity of the solutions, a change occurs from the coefficient of a feature attaining zero, or becoming nonzero. In both cases, the active set is changed by only one element (except for degenerate cases that will be discussed in section 3.1), and the change is closely related to the conditions for activation/inactivation in the active set procedure. This consideration leads to the homotopy algorithm presented in section 2.3.

After having established the equivalence and explicitated the relation between the two formulations of the LASSO, we can examine how the active set descent method can be applied to the regularized LASSO.

2.2.3 Active set descent for the regularized LASSO

The application of active set descent to the regularized LASSO was considered by their authors in remark 6 of (Osborne et al., 2000b) that we partially quote here:

“...In principle, our algorithm can also be used to solve [the regularized LASSO]. In this case it would be necessary to find that value of t for which the corresponding Lagrange multiplier is equal to the smoothing parameter λ in [the regularized LASSO]. This could be done within a further loop, either by performing a grid search or using a Newton-Raphson algorithm...”

The exposition of the active set descent that was just conducted lets a much simpler alternative appear, that need not consider the relation between t and λ . The regularized formulation is a simpler problem of minimizing a strictly convex function, without an inequality constraint. The same positivity trick makes it a canonical quadratic program, the only differences with the previous problem being that:

- the objective function is $\|\mathbf{y} - \mathbf{X}^T\boldsymbol{\beta}\|_2^2 + \lambda\mathbf{w}^T\boldsymbol{\beta}$ instead of $\|\mathbf{y} - \mathbf{X}^T\boldsymbol{\beta}\|_2^2$,
- no oblique inequality is involved.

The new objective is a strictly convex quadratic function and therefore a valid candidate to the direct application of the corresponding active set descent method, which gives Algorithm 6, in which, comparing to Algorithm 5, the computation of λ_σ at each step is simply replaced by the use of the given parameter λ . This algorithm was introduced in (Loth and Preux, 2010), under the name *Iso-Regularization Descent*. This name was given in contrast with *Iso-Constraint Descent* by which we denoted the constrained-LASSO version, that descends on the surface of the constraint polytope. We abandon these terms, in favour of the general name of the algorithm (ASD), the regularized form being somehow the implicit, standard, form of the LASSO.

As we have seen in the previous chapter and in section 2.2.2.3, the regularized formulation is a more practical one, for the working bounds of its parameter are known in advance. Moreover, the active set method can be used to solve it, with an algorithm that is simpler to state and uses less computations; removing the computations of successive Lagrange multipliers not only decreases execution time, but also lessens the risks related to arithmetic computational precision.

Consequently, from now on, we abandon the constrained formulation for the remaining of this thesis, and focus on the regularized problem:

$$\text{minimize } \mathcal{L}(\sigma, \boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}_\sigma^T\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_{\mathbf{w}_\sigma}. \quad (2.16)$$

This switch is consistent with the evolution of the scientific publications on the LASSO, from its –constrained– formulation by (Tibshirani, 1996), to the growing

Algorithm 6 active set descent for the regularized LASSO

Input: input vector $\mathbf{x} \in \mathcal{X}^n$, response vector $\mathbf{y} \in \mathbb{R}^n$, feature dictionary $\mathcal{D} \subset \mathbb{R}^{\mathcal{X}}$, penalization function $w : \mathcal{D} \rightarrow \mathbb{R}_+$, regularization parameter $\lambda \in \mathbb{R}_+$

Output: $\text{LASSO}_r(\mathbf{x}, \mathbf{y}, \mathcal{D}, \mathbf{w}, \lambda) = \arg \min_{(\sigma, \boldsymbol{\beta}) \in 2^{\mathcal{D}} \times \mathbb{R}^{\mathcal{D}}} \|\mathbf{y} - \mathbf{X}_\sigma \boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_{\mathbf{w}_\sigma}$

Definition: For a feature function $\phi \in \mathcal{D}$, the bold notation denotes the application of ϕ to the input vector \mathbf{x} : $\boldsymbol{\phi} = \phi(\mathbf{x}) = (\phi(x_1), \dots, \phi(x_n))^T$.

```

1:  $\sigma \leftarrow \{\}$ ,  $\mathbf{X} \leftarrow []$ ,  $\boldsymbol{\theta} \leftarrow ()$ 
2: loop
3:   repeat
4:      $\bar{\boldsymbol{\beta}} \leftarrow (\mathbf{X}\mathbf{X}^T)^{-1}(\mathbf{X}\mathbf{y} - \lambda\boldsymbol{\theta})$ 
5:      $(\gamma, \phi) \leftarrow \min, \arg \min_{\phi \in \sigma} \left( \frac{\beta_\phi}{\beta_\phi - \bar{\beta}_\phi} \right)_{\geq 0}$ 
6:     if  $\gamma \leq 1$  then
7:        $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \gamma(\bar{\boldsymbol{\beta}} - \boldsymbol{\beta})$ 
8:        $\sigma \leftarrow \sigma \setminus \{\phi\}$ 
9:       shrink  $\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\theta}$  accordingly
10:    end if
11:    until  $\gamma \geq 1$ 
12:     $\boldsymbol{\beta} \leftarrow \bar{\boldsymbol{\beta}}$ 
13:     $\phi \leftarrow \arg \min_{\phi \in \mathcal{D}} [g(i) = \lambda w(\phi) - |\boldsymbol{\phi}(\mathbf{y} - \mathbf{X}^T \boldsymbol{\beta})|]$ 
14:    if  $g(i) < 0$  then
15:       $\sigma \leftarrow \sigma \cup \{\phi\}$ 
16:      extend  $\mathbf{X}$  by row  $\boldsymbol{\phi}$ ,  $\boldsymbol{\beta}$  by 0, and  $\boldsymbol{\theta}$  by  $w(\phi) \text{sign}(\boldsymbol{\phi}(\mathbf{y} - \mathbf{X}^T \boldsymbol{\beta}))$ 
17:    else
18:      return  $(\sigma, \boldsymbol{\beta})$ 
19:    end if
20: end loop

```

interest for its regularized version and associated algorithms. The breakthrough in this popularity is the publication in (Efron et al., 2004) of the least angle regression method (LAR or LARS) that unifies several feature selection procedures, among which one (LAR-LASSO) that solves the regularized LASSO for all values of its parameter. This algorithm was initially presented – as the *homotopy* method – together with the active set method in (Osborne et al., 2000a; Osborne et al., 2000b), for the constrained formulation. The transposition to the regularized LASSO is of the same nature as for the active set method: it can be resumed to working directly on λ rather than computing it as a function of t . We present this algorithm in the next section, as a derivation of the active set method; this reformulated presentation aims at the same benefits as that of the active set method: generalization of the LASSO settings, simplicity, and enlightenment of some possibilities or issues –and their resolutions– of the algorithm.

2.3 The homotopy method

The idea of the homotopy method is to follow the path of solutions defined by the interval $[0, \lambda_0]$ on which the parameter λ is meaningful. This path, discussed in section 2.2.2.3 is referred to as the regularization path. The name *homotopy*, given to this method when introduced in (Osborne et al., 2000a), is a term mostly used in topology, to designate a continuous deformation between two functions f and g , that is a continuum of functions h_l (evolving continuously with $l \in [0, 1]$) such that $h_0 = f$ and $h_1 = g$. LASSO estimators evolve continuously with the parameter λ , hence the idea of following this homotopy rather than solving single problems independently.

The intuition of the algorithm can be derived in a natural fashion from the active set descent algorithm. When a LASSO problem for a given value of λ has been solved by the latter, together with the solution $(\sigma, \beta_\lambda = \beta^* - \lambda \Delta \beta)$ comes the solutions of problems for the whole interval of parameters where the optimal active set is σ . This interval is defined by the fact that the termination conditions of the algorithm are met, that is:

- β_λ is feasible, i.e. matches the sign of θ ,
- no inactive feature has a negative derivative on the loss function.

The linearity in λ of the solution and the gradient of the loss makes it easy to compute the bounds of this interval, and with this computation is determined which of the feature violates the optimality conditions. We then know that beyond the bounds of the interval, a supplementary step is required for the active set method that consists in activating or inactivating the corresponding feature. The special case of simultaneous violations occurring at a bound will be addressed in chapter 3, and we begin by expliciting the computation of the bounds, focusing on the lower bound (we track the evolutions as λ decreases). It consists in determining :

- the largest value $\lambda_- < \lambda$ for which the sign of an active features becomes zero (thus changes beyond),
- the largest value $\lambda_+ < \lambda$ for which an inactive feature shows a zero (thus negative beyond) gradient.

Then the largest value among λ_- and λ_+ gives the first value where the active set is changing, when decreasing the regularization parameter.

2.3.1 Computing the first unfeasibility point

Let λ denote an arbitrary point of the interval, and λ_- the closest lower value beyond which an active feature changes sign.

The solution evolves linearly with λ , following the expression

$$\beta_\lambda = \beta^* - \lambda \Delta \beta. \quad (2.17)$$

The point where a coefficient is zeroed is thus given by

$$\lambda_-(\phi) = \frac{\beta_\phi^*}{\Delta \beta_\phi}.$$

The fact that a coefficient increases in absolute value with λ decreasing – thus not changing sign – translates in $\lambda_-(\phi)$ being greater than λ . Similarly, for a newly activated feature, that has a coefficient zero, λ_- is equal to λ . This gives:

$$\lambda_- = \max_{\phi \in \sigma} \left(\frac{\beta_\phi^*}{\Delta \beta_\phi} \right)_{< \lambda}. \quad (2.18)$$

However, it is simpler to discard directly the features that do not change sign when decreasing λ . For this, it suffices to consider β_ϕ^* , which is the ultimate value of β_ϕ , for $\lambda = 0$. A feature must be considered for the next unfeasibility point only if this ultimate value is unfeasible, i.e. its sign differs from θ_ϕ . Those features, and only those, have a value of λ_- in $[0, \lambda]$, and the largest is to be retained. Another benefit of this preliminary restriction is that it allows to handle features that have a zero coefficient. In normal circumstances, their coefficient evolve in a correct direction once they are activated, and they need not be considered, whence the criterion ($< \lambda$) in (2.18). However, in degenerate cases that will be discussed in the following chapter, such a newly activated feature might need to be inactivated immediately, its coefficient evolving in the wrong direction after the –immediate– activation of another feature. The difference between the two cases is made by the β^* criterion. Hence the first unfeasibility point can be obtained by

$$\lambda_- = \max_{\substack{\phi \in \sigma \\ \text{sign}(\beta_\phi^*) \neq \text{sign}(\theta_\phi)}} \frac{\beta_\phi^*}{\Delta \beta_\phi}. \quad (2.19)$$

An other possible criterion is simply to discard the features of which the coefficient's evolution Δ_{β_ϕ} has an opposite sign as that of θ_ϕ , thus the coefficient increases in absolute value as λ decreases (eq. (2.17)).

$$\lambda_- = \max_{\substack{\phi \in \sigma \\ \text{sign}(\Delta_{\beta_\phi}) \neq \text{sign}(\theta_\phi)}} \frac{\beta_\phi^*}{\Delta_{\beta_\phi}}. \quad (2.20)$$

This filter leaves features of which the coefficient zeroes in $(-\infty, \lambda]$, rather than features zeroing in $[0, \lambda]$ for (2.19).

2.3.2 Computing the first insufficiency point

A feature ϕ can be activated with a positive or negative coefficient, and it is again easier to consider separately $+\phi$ and $-\phi$ (noting s the sign), both with a positive coefficient, making the loss differentiable, with a derivative equal to

$$\frac{\partial \mathcal{L}(\boldsymbol{\beta})}{\partial \beta_{s\phi}} = -\phi^\top (\mathbf{y} - \mathbf{X}_\sigma^\top \boldsymbol{\beta}) + s\lambda w(\phi)$$

σ becomes insufficient when activating a feature can decrease the loss, that is when its derivative with respect to that feature becomes negative. Again, since all these derivatives are non-negative in the interval and evolve linearly, this violation occurs beyond the point λ_+ where a derivative is zeroed. For a given feature $s\phi$, violation occurs at point $\tilde{\lambda}_+^s$ characterized by

$$-s\phi^\top (\mathbf{y} - \mathbf{X}_\sigma^\top \boldsymbol{\beta}(\tilde{\lambda}_+^s)) + \tilde{\lambda}_+^s w(\phi) = 0, \quad (2.21)$$

or equivalently

$$\phi^\top (\mathbf{y} - \mathbf{X}_\sigma^\top \boldsymbol{\beta}(\tilde{\lambda}_+^s)) - s\tilde{\lambda}_+^s w(\phi) = 0, \quad (2.22)$$

$$\begin{aligned} (2.22) &\iff \phi^\top (\mathbf{y} - \mathbf{X}_\sigma^\top (\boldsymbol{\beta}^* - \tilde{\lambda}_+^s \boldsymbol{\Delta} \boldsymbol{\beta})) - s\tilde{\lambda}_+^s w(\phi) = 0 \\ &\iff \phi^\top \underbrace{(\mathbf{y} - \mathbf{X}_\sigma^\top \boldsymbol{\beta}^*)}_{\mathbf{r}^*} + \tilde{\lambda}_+^s \phi^\top \underbrace{(\mathbf{X}_\sigma^\top \boldsymbol{\Delta} \boldsymbol{\beta})}_{\boldsymbol{\Delta}_\mathbf{r}} - s\tilde{\lambda}_+^s w(\phi) = 0 \\ &\iff \tilde{\lambda}_+^s = \frac{\phi^\top \mathbf{r}^*}{s w(\phi) - \phi^\top \boldsymbol{\Delta}_\mathbf{r}} \end{aligned}$$

A naive procedure consists in computing both $\tilde{\lambda}_+^+$ and $\tilde{\lambda}_+^-$ for each inactive feature and retain the greater of those that are less than λ , but the computation can be simplified by the property that only $s^* \phi$ can require activation when decreasing λ , where $s^* = \text{sign}(\phi^\top \mathbf{r}^*)$ and activation occurs for $\lambda = \frac{\phi^\top \mathbf{r}^*}{s^* w(\phi) - \phi^\top \boldsymbol{\Delta}_\mathbf{r}}$.

Proof. By definition, if σ is optimal for parameter λ , and $\phi \notin \sigma$, both derivatives are positive:

$$\begin{cases} -\phi^\top \mathbf{r}^* + \lambda \phi^\top \Delta_{\mathbf{r}} + \lambda w(\phi) \geq 0 \\ \phi^\top \mathbf{r}^* - \lambda \phi^\top \Delta_{\mathbf{r}} + \lambda w(\phi) \geq 0 \end{cases}$$

When decreasing λ , these derivatives evolve linearly to the limit values (for $\lambda = 0$) of $-\phi^\top \mathbf{r}^*$ and $\phi^\top \mathbf{r}^*$, of which only one has become negative. Hence the positivity of $\phi^\top \mathbf{r}^*$ implies that only $+\phi$ requires activation, for $\lambda < \tilde{\lambda}_+^+$, and its negativity implies that only $-\phi$ requires activation, for $\lambda < \tilde{\lambda}_+^-$. \square

This gives

$$\lambda_+ = \max_{\phi \in \mathcal{D}-\sigma} \frac{\phi^\top \mathbf{r}^*}{\text{sign}(\phi^\top \mathbf{r}^*)w(\phi) - \phi^\top \Delta_{\mathbf{r}}} \quad (2.23)$$

This maximization can be performed over the whole set \mathcal{D} , since active features always give, by construction of the solution, a zero derivative. This implies that if $\phi \in \sigma$, $\phi^\top \mathbf{r}^*$, as the derivative at $\lambda = 0$, is equal to zero, and does not “compete” in the maximization. This property does not hold in the formulation of (Efron et al., 2004), where the smallest change in λ is computed, rather than the new λ itself, which leads to the opposite effect of always selecting already active features if they were to be considered. Although this may seem a minor point, it is useful when dealing with large feature sets, as will be explained in chapter 3.

2.3.3 Jumping to the next break point

With these two formulae (2.19,2.23), given the solution (σ, β) for parameter λ , we can compute the parameter λ' down to which σ remains optimal, and also which condition is violated beyond it, and the corresponding new active set. We can then consider the interval on which this new set is optimal, by updating the vectors related to the active set, and repeating the computation of the lower bound, and iterate until the desired value of λ , or of an other criterion, is reached, and at most until the unconstrained least square at $\lambda = 0$. This procedure can be used in both directions of the regularization path, with the following corrections for increasing λ :

- the only sign with which a feature may be activated when increasing λ is the opposite of that when decreasing λ , from the linearity of the correlation wrt. λ ; thus $\text{sign}(\phi^\top \mathbf{r}^*)$ must be replaced by $-\text{sign}(\phi^\top \mathbf{r}^*)$ in (2.23);
- when computing the zeroing point λ_- of a coefficient in (2.20), we discard coefficients that do not zero in $(-\infty, \lambda]$, i.e. that zero in $[\lambda, \infty)$, by the condition $\text{sign}(\Delta_{\beta_\phi}) \neq \text{sign}(\theta_\phi)$. The opposite filter must be used when increasing λ , that is $\text{sign}(\Delta_{\beta_\phi}) = \text{sign}(\theta_\phi)$.

Therefore, the general formula for the first unfeasibility and first insufficiency points are

$$\lambda_+ = \max_{\phi \in \mathcal{D}^{-\sigma}} \frac{\phi^\top \mathbf{r}^*}{-s_p \text{sign}(\phi^\top \mathbf{r}^*) w(\phi) - \phi^\top \Delta \mathbf{r}} \quad (2.24)$$

$$\lambda_- = \max_{\substack{\phi \in \sigma \\ s_p \text{sign}(\Delta_{\beta_\phi}) = \text{sign}(\theta_\phi)}} \frac{\beta_\phi^*}{\Delta_{\beta_\phi}} \quad (2.25)$$

where s_p is the sign of the evolution of λ that is considered (-1 for λ decreasing).

It is natural, though, since a solution is needed as a starting point, to initiate the homotopy procedure with $\lambda = \infty$ and the corresponding trivial solution of an empty active set, and iterate in the decreasing direction. The algorithm that computes the whole regularization path is given in Algorithm 7.

This formulation offers the benefit of giving a directly implementable algorithm with a low number of arithmetic operations. This simplifies its understanding and analyses, and also, naturally, its efficiency.

Regarding this point, a costly part and source of optimization in these algorithms is the matrix inversion appearing in the computations of β^* and $\Delta \beta$. Contrary to (Osborne et al., 2000a), we treat separately the main algorithm and the optimization of this part, that somewhat belong to different algorithmic levels. This optimization is addressed in the section 2.4.1.

2.4 Complexity

Let us, more generally, analyse the complexity of the ASD and homotopy algorithms, as described in algorithms 6 and 7.

The main three tasks repeated in the main loop of both algorithms are:

zeroing identify among the active features the first of which the coefficient is zeroed when following the descent path (ASD) or decreasing λ (homotopy),

over-correlation identify among the inactive features which one is the most over-correlated (ASD) or is the first to reach equi-correlation when decreasing λ (homotopy),

update computing a solution associated to the active set each time it has changed.

Let us note, as previously,

- n the number of observations,
- p the cardinality of \mathcal{D} (number of features, although it may not be finite),
- k the number of active features at the step of the main loop that we consider.

Algorithm 7 Homotopy method for the regularized LASSO

Input: input vector $\mathbf{x} \in \mathcal{X}^n$, response vector $\mathbf{y} \in \mathbb{R}^n$, feature dictionary $\mathcal{D} \subset \mathbb{R}^{\mathcal{X}}$, penalization function $w : \mathcal{D} \rightarrow \mathbb{R}_+$,

Output: For all $\lambda \in \mathbb{R}_+$,
 $\text{LASSO}_r(\mathbf{x}, \mathbf{y}, \mathcal{D}, w, \lambda) = \arg \min_{(\sigma, \beta) \in 2^{\mathcal{D}} \times \mathbb{R}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \beta\|_2^2 + \lambda \|\beta\|_{\mathbf{w}_\sigma}$

Definition: For a feature function $\phi \in \mathcal{D}$, the bold notation denotes the application of ϕ to the input vector \mathbf{x} : $\boldsymbol{\phi} = \phi(\mathbf{x}) = (\phi(x_1), \dots, \phi(x_n))^\top$.

- 1: $\lambda \leftarrow \infty, \sigma \leftarrow \{\}, \mathbf{X} \leftarrow [], \boldsymbol{\theta} \leftarrow ()$
- 2: **loop**
- 3: $\boldsymbol{\beta}^* \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1} \mathbf{X}\mathbf{y}$
- 4: $\boldsymbol{\Delta}\boldsymbol{\beta} \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1} \boldsymbol{\theta}$
- 5: $\mathbf{r}^* \leftarrow \mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}^*$
- 6: $\boldsymbol{\Delta}_r \leftarrow \mathbf{X}^\top \boldsymbol{\Delta}\boldsymbol{\beta}$
- 7: $(\lambda_-, \phi_-) \leftarrow \max_{\substack{\phi \in \sigma \\ \text{sign}(\beta_\phi^*) \neq \text{sign}(\theta_\phi)}} \arg \max \frac{\beta_\phi^*}{\Delta \beta_\phi} \quad \triangleright \text{sec. 2.3.1}$
- 8: $(\lambda_+, \phi_+) \leftarrow \max_{\phi \in \mathcal{D} \setminus \sigma} \arg \max \frac{\boldsymbol{\phi}^\top \mathbf{r}^*}{\text{sign}(\boldsymbol{\phi}^\top \mathbf{r}^*) w(\phi) - \boldsymbol{\phi}^\top \boldsymbol{\Delta}_r} \quad \triangleright \text{sec. 2.3.2}$
- 9: $\lambda' \leftarrow \max(\lambda_-, \lambda_+, 0)$
- 10: **output** $\forall l \in [\lambda', \lambda), \text{LASSO}_r(\mathbf{x}, \mathbf{y}, \mathcal{D}, w, l) \leftarrow (\sigma, \boldsymbol{\beta}^* - l \boldsymbol{\Delta}\boldsymbol{\beta})$
- 11: **if** $\lambda' = 0$ **then terminate**
- 12: **else if** $\lambda' = \lambda_-$ **then** \triangleright inactivation
- 13: $\sigma \leftarrow \sigma \setminus \{\phi_-\}$
- 14: shrink $\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\theta}$ accordingly
- 15: **else if** $\lambda' = \lambda_+$ **then** \triangleright activation
- 16: $\sigma \leftarrow \sigma \cup \{\phi_+\}$
- 17: extend \mathbf{X} by row $\boldsymbol{\phi}$, $\boldsymbol{\beta}$ by 0, and $\boldsymbol{\theta}$ by $\text{sign}(\boldsymbol{\phi}^\top \mathbf{r}^*) w(\phi)$
- 18: **end if**
- 19: $\lambda \leftarrow \lambda'$
- 20: **end loop**

The complexity of the *zeroing* task is $O(1) * k = O(k)$.

The *over-correlation* task involves the computation, for each inactive feature, of its correlation to the residual, possibly decomposed into the least-squares residual and the “delta-residual”. Given the residual, the complexity is then $O(n^2) * (p - k)$. k evolving from 0 to the final sparsity of the solution, typically small w.r.t. p , one can simplify this complexity to $O(n^2 p)$.

The *update* task, following each change in the active set, involves the extension or shrinkage of \mathbf{X} , $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$, and the update of the residual, yielding a complexity of $O(n)$. Most importantly, a least-squares operation is performed to compute $\hat{\boldsymbol{\beta}}$ or $\boldsymbol{\beta}^*$ and $\Delta\boldsymbol{\beta}$. An implementation that strictly follows the algorithms as they are exposed here requires the computation of the Gram product $\mathbf{X}\mathbf{X}^\top$ ($O(nk^2)$), its inversion ($O(p^{2.376})$ to $O(k^3)$, let us consider the latter), and one or two multiplications with a vector ($O(k^2)$). This gives an overall complexity of $O(k^3 + nk^2)$ for the *update* task. However, \mathbf{X} is only modified by one row between each iteration of the main loop, and this allows to reduce this complexity of one order in k ($O(k^2 + nk)$), by using dedicated methods for solving *sequential* least-squares problems, as explained in section 2.4.1. With a slight simplification, the complexity of one iteration of the main loop is thus $O(n^2 p + nk^2 + k^3)$. The simplification lies in the fact that the two algorithms do not perform the same number of tasks in each iteration:

- the homotopy tests both zeroing and over-correlation once, and performs one shrinkage *or* expansion, thus one update;
- the active set descent may perform several zeroings (and consequent shrinkage and updates) before testing over-correlation and activating the corresponding feature.

Thus, the complexity in $n^2 p + nk^2 + k^3$ is exact for the homotopy, but underestimated for the active set descent. However, the number of zeroings in the latter is inestimable in a similar and related way as the number of iterations of the main loop needed to converge to the LASSO solution. The problem is the same as that of the complexity of the simplex algorithm. Therefore, in order to estimate the complexity of the algorithms, it should simply be noted that it is dominated by the over-correlation and update tasks ($O(n^2 p)$ and $O(nk^2 + k^3)$ respectively), which are performed an inestimable number of times, that can be studied from empirical evidences. A noteworthy point preliminary to such a study is that those two numbers are differently related in both algorithms: in the homotopy, every change in the active set (activation or inactivation) is preceded by an over-correlation test, whereas in the active set descent, every such test is followed by an activation.

In order to estimate how these complexities behave in practice, we have thus run experiments to monitor the number of both tasks needed to reach a LASSO solution for both algorithms (homotopy and ASD for the regularized LASSO). The components of \mathbf{X} were drawn independently from $\mathcal{N}(0, 1)$; \mathbf{y} was built by

a combination \mathbf{f} of k of these features, k going from 1 to $0.8 \max(n, p)$, with coefficients independently drawn from the uniform distribution in $[-1, 1]$, and a noise term was added to \mathbf{f} , drawn independently for each observation from $\mathcal{N}(0, 0.01 * \|\mathbf{f}\|_2^2/n)$. On 100 different such data for each k , we have run the homotopy algorithm until the LASSO solution included k features, and then ASD was run with the corresponding value of λ , starting from an empty active set. The number of over-correlation tests and of active set updates were counted, and figure 2.4 shows the average number of tasks required by both algorithms as a function of the cardinality of the active set of the LASSO solution, for $n = 150, p = 1000, \rho = 0$, averaged on 1000 runs. Other settings give similar results, that is the number of over-correlation tests is consistently lower for ASD than for homotopy, though not from a great amount, while the number of updates tends to be the same for both algorithms as k approaches $2n/3$, and greater for ASD beyond. Globally, the number of steps needed to reach a k -sparse LASSO solution remains in the order of k , which gives both algorithms an overall approximate complexity of

$$O(n^2pk + nk^3 + k^4)$$

to compute a LASSO solution selecting k features out of p on n observations.

2.4.1 Sequential least-squares

As noted in the introduction of this section, although the inverses of the Gram matrices $\mathbf{X}\mathbf{X}^\top$ appear in the exposition of algorithms, actual inversions need not be done in practice. On the contrary, there are several reasons to compute solutions by mean of a decomposition of the Gram matrix: numerical stability, degeneracy detection, and – most of all – a low computational cost for updating the decomposition when the active set is changed by one element. The principle of solving linear systems using a matrix decomposition (QR or Cholesky) is well known and commonly used; the update of such decompositions when the coefficient matrix is added a rank one matrix is also documented; let us however recall those for the Cholesky decomposition, before exposing how to update the decomposition when a feature (row of \mathbf{X}_σ) is added or removed, which we could not find in the literature or Internet and, though not very complex, is worth an explanation. The choice of Cholesky over QR is motivated by the fact that the use of the latter in the homotopy algorithm is already explained in (Osborne et al., 2000a), and also by the possibility it offers to handle online learning, as will be seen in next chapter.

2.4.2 Cholesky decomposition

Using notations that are consistent with the previous ones, let \mathbf{X} be a $p \times n$ matrix and \mathbf{b} a vector of length p for which we wish to compute

$$\mathbf{v} = (\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{b} \tag{2.26}$$

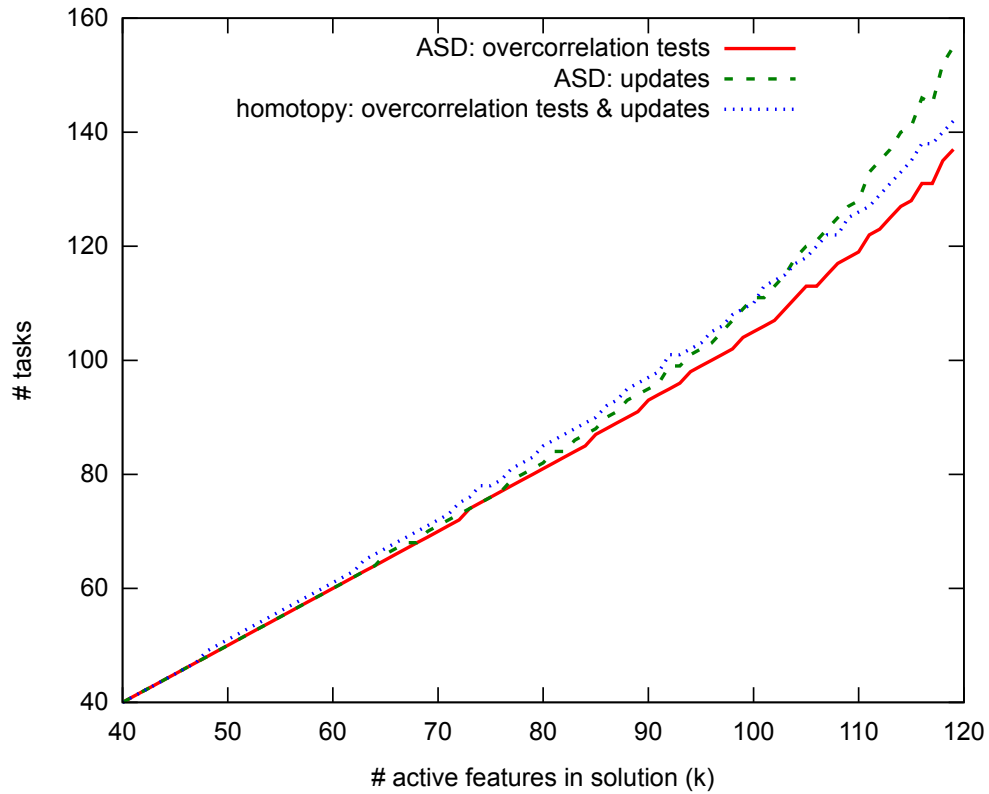


Figure 2.4: Average numbers of the most costly tasks required to reach a LASSO solution, as a function of the cardinality of the solution (k), for random and unstructured problems.

Number of runs for each k : 100; number of observations: $n = 150$; number of features: $p = 1000$.

For both algorithms, the number of steps remains in the order of k . The most costly task (for $p > n$) of over-correlation test ($O(n^2p)$) is performed consistently less in the ASD algorithm, while the update task ($O(k^2 + nk)$) tends to be performed more often by ASD when the LASSO solution approaches the least-squares solution.

or equivalently solve

$$\mathbf{X}\mathbf{X}^\top \mathbf{v} = \mathbf{b} \quad (2.27)$$

for \mathbf{v} .

The Cholesky decomposition of $\mathbf{X}\mathbf{X}^\top$ consists in finding a lower triangular matrix \mathbf{L} with positive diagonal entries such that

$$\mathbf{X}\mathbf{X}^\top = \mathbf{L}\mathbf{L}^\top \quad (2.28)$$

(2.27) is then equivalent to the system

$$\begin{cases} \mathbf{L}\mathbf{y} = \mathbf{b} \\ \mathbf{L}^\top \mathbf{v} = \mathbf{y} \end{cases} \quad (2.29)$$

of which the two parts can be efficiently solved by back substitutions, thanks to the triangular form of \mathbf{L} .

2.4.3 Cholesky rank one update

Let \mathbf{u} be a vector of length p . The outer product $\mathbf{u}\mathbf{u}^\top$ is a matrix of rank one to be added to $\mathbf{X}\mathbf{X}^\top$, and we wish to compute the decomposition $\mathbf{L}'\mathbf{L}'^\top$ of the resulting matrix from that of the original one:

$$\mathbf{L}'\mathbf{L}'^\top = \mathbf{L}\mathbf{L}^\top + \mathbf{u}\mathbf{u}^\top \quad (2.30)$$

Methods for computing \mathbf{L}' from \mathbf{L} can be found in (Gill et al., 1974; Golub and Saunders, 1969; Seeger, 2008), and are implemented in most algebra packages/software, but we give an extensive explanation of the common method here, in order to provide a more readable exposition, that also serves as a documentation of our implementations that do not rely on these existing packages. In order to clarify the explanation and without loss of generality, we use an example with $p = 4$ and represent only the nonzero parts of the matrices.

Let us first note that

and \mathbf{L}' is thus such that

or equivalently

$$\begin{array}{|c|c|} \hline \mathbf{u} & \mathbf{L} \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{u}^\top \\ \hline \mathbf{L}^\top \\ \hline \end{array} = \begin{array}{|c|c|} \hline \mathbf{0} & \mathbf{L}' \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{0} \\ \hline \mathbf{L}'^\top \\ \hline \end{array}.$$

Let us first transform \mathbf{u} and the first column of \mathbf{L} (denoted by \mathbf{a}), so that the first element of \mathbf{u} is zeroed, and the Gramian as well as the positivity of the diagonal are preserved.

Since

$$\begin{array}{|c|c|} \hline \mathbf{u} & \mathbf{a} \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{L}_{\bar{a}} \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{u}^\top \\ \hline \mathbf{a}^\top \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{L}_{\bar{a}}^\top \\ \hline \end{array} = \begin{array}{|c|c|} \hline \mathbf{u} & \mathbf{a} \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{u}^\top \\ \hline \mathbf{a}^\top \\ \hline \end{array} + \begin{array}{|c|c|} \hline \mathbf{L}_{\bar{a}} \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{L}_{\bar{a}}^\top \\ \hline \end{array},$$

we only need to preserve the Gramian of $(\mathbf{u}|\mathbf{a})$ (the juxtaposition of \mathbf{u} and \mathbf{a} as seen above). A transformation that preserves this product is a rotation, obtained by multiplying $(\mathbf{u}|\mathbf{a})$ by a matrix $\begin{pmatrix} c & -s \\ s & c \end{pmatrix}$, with $c^2 + s^2 = 1$:

$$(\mathbf{u}|\mathbf{a}) \begin{pmatrix} c & -s \\ s & c \end{pmatrix} \begin{pmatrix} c & s \\ -s & c \end{pmatrix} (\mathbf{u}|\mathbf{a})^\top = (\mathbf{u}|\mathbf{a}) \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} (\mathbf{u}|\mathbf{a})^\top = (\mathbf{u}|\mathbf{a})(\mathbf{u}|\mathbf{a})^\top$$

This transformation must zero the first element u_1 of \mathbf{u} , hence we must solve

$$cu_1 + sa_1 = 0 \tag{2.31}$$

$$-su_1 + ca_1 > 0 \tag{2.32}$$

$$c^2 + s^2 = 1 \tag{2.33}$$

of which the solution is

$$c = \frac{a_1}{\sqrt{u_1^2 + a_1^2}}$$

$$s = -\frac{u_1}{\sqrt{u_1^2 + a_1^2}}$$

Thus, this transformation, referred to as a *Givens rotation* or *Householder transformation*, gives vectors \mathbf{u}' and \mathbf{a}' such that

$$\begin{array}{|c|c|} \hline \mathbf{0} & \mathbf{u}' \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{a}' \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{L}_{\bar{a}} \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{u}'^\top & \mathbf{0} \\ \hline \mathbf{a}'^\top \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{L}_{\bar{a}}^\top \\ \hline \end{array} = \begin{array}{|c|c|} \hline \mathbf{u} & \mathbf{a} \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{L}_{\bar{a}} \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{u}^\top \\ \hline \mathbf{a}^\top \\ \hline \end{array} \begin{array}{|c|c|} \hline \mathbf{L}_{\bar{a}}^\top \\ \hline \end{array}.$$

The same operation can be repeated to zero the first element of \mathbf{u}' by rotating \mathbf{u}' and the second column of \mathbf{L} , and so on until \mathbf{u} is zeroed and \mathbf{L} has been transformed into the objective matrix \mathbf{L}' .

2.4.4 Feature addition

Let us now consider the addition of a feature to the active set in the algorithms of the previous section, that is the addition of a row \mathbf{u}^\top to the matrix \mathbf{X} . The ordering of features in the active set being arbitrary, we assume this row is added at the bottom. Let \mathbf{X}' be the augmented matrix:

$$\mathbf{X}' = \left(\begin{array}{c} \mathbf{X} \\ \hline \mathbf{u}^\top \end{array} \right),$$

then

$$\mathbf{X}'\mathbf{X}'^\top = \left(\begin{array}{c|c} \mathbf{X}\mathbf{X}^\top & \mathbf{X}\mathbf{u} \\ \hline \mathbf{u}^\top\mathbf{X}^\top & \mathbf{u}^\top\mathbf{u} \end{array} \right).$$

Since only the additional row and column of the Gramian change, only the corresponding additional row of \mathbf{L} need be computed. If we note w the new bottom-right entry and \mathbf{v} the remaining of the new row:

$$\mathbf{L}' = \left(\begin{array}{c|c} \mathbf{L} & \mathbf{0} \\ \hline \mathbf{v}^\top & w \end{array} \right),$$

then we have

$$\mathbf{L}'\mathbf{L}'^\top = \left(\begin{array}{c|c} \mathbf{L}\mathbf{L}^\top & \mathbf{L}\mathbf{v} \\ \hline \mathbf{v}^\top\mathbf{L}^\top & \mathbf{v}^\top\mathbf{v} + w^2 \end{array} \right).$$

So \mathbf{v} is obtained by solving

$$\mathbf{L}\mathbf{v} = \mathbf{X}\mathbf{u}, \quad (2.34)$$

that is computing $\mathbf{X}\mathbf{u}$, and using back substitutions with \mathbf{L} , and w is given by

$$w = \sqrt{\mathbf{u}^\top\mathbf{u} - \mathbf{v}^\top\mathbf{v}} \quad (2.35)$$

2.4.5 Feature subtraction

For the removal of a feature, we cannot assume it has a specific position in \mathbf{X} , and will consider an arbitrary position i , and use $i = 3$ in the illustrations.

When row i is removed from \mathbf{X} , the i -th row and i -th column are removed from $\mathbf{X}\mathbf{X}^\top$, leaving the rest unchanged. Let us again note \mathbf{X}' the truncated matrix, $\mathbf{G}' = \mathbf{X}'\mathbf{X}'^\top$ the new Gramian, and $\mathbf{G}_{11}, \mathbf{G}_{12}, \mathbf{G}_{22}$ the sub-matrices delimited by the removed row and column:

$$\mathbf{X}\mathbf{X}^\top = \mathbf{L}\mathbf{L}^\top = \begin{array}{|c|c|c|} \hline \mathbf{G}_{11} & & \mathbf{G}_{21}^\top \\ \hline & & \\ \hline \mathbf{G}_{21} & & \mathbf{G}_{22} \\ \hline \end{array}, \quad \mathbf{X}'\mathbf{X}'^\top = \mathbf{L}'\mathbf{L}'^\top = \begin{array}{|c|c|} \hline \mathbf{G}_{11} & \mathbf{G}_{21}^\top \\ \hline \mathbf{G}_{21} & \mathbf{G}_{22} \\ \hline \end{array}$$

\mathbf{L} must also be truncated by its i -th row and column; let us similarly name the sub-matrices of \mathbf{L} and \mathbf{L}' , as well as the part of the i -th column of \mathbf{L} below the diagonal:

$$\mathbf{L} = \begin{array}{|c|c|c|c|} \hline & & & \\ \hline & \mathbf{L}_{11} & & \\ \hline & & & \\ \hline & \mathbf{L}_{21} & \mathbf{l}_i & \mathbf{L}_{22} \\ \hline \end{array}, \quad \mathbf{L}' = \begin{array}{|c|c|} \hline & \mathbf{L}'_{11} \\ \hline & \mathbf{L}'_{21} & \mathbf{L}'_{22} \\ \hline \end{array}$$

Since \mathbf{G}_{11} must be equal to both $\mathbf{L}_{11}\mathbf{L}_{11}^\top$ and $\mathbf{L}'_{11}\mathbf{L}'_{11}^\top$, we have

$$\mathbf{L}'_{11} = \mathbf{L}_{11}.$$

Consequently, we also have

$$\mathbf{L}'_{21} = \mathbf{L}_{21},$$

because \mathbf{G}_{21} must be equal to both $\mathbf{L}_{11}\mathbf{L}_{21}^\top$ and $\mathbf{L}_{11}\mathbf{L}'_{21}^\top$.

Thus, only \mathbf{L}'_{22} differs from the original \mathbf{L}_{22} , and can be computed from the fact that \mathbf{G}_{22} must be equal to both $(\mathbf{l}_i|\mathbf{L}_{22})(\mathbf{l}_i|\mathbf{L}_{22})^\top$ and $\mathbf{L}'_{22}\mathbf{L}'_{22}^\top$. This equation is a rank-one update, that can be computed as was explained previously:

$$\begin{array}{|c|c|} \hline \mathbf{l}_i & \mathbf{L}_{22} \\ \hline \end{array} \begin{array}{|c|} \hline \mathbf{l}_i^\top \\ \hline \mathbf{L}_{22}^\top \\ \hline \end{array} = \begin{array}{|c|c|} \hline & \mathbf{L}'_{22} \\ \hline \end{array} \begin{array}{|c|} \hline \mathbf{L}'_{22}^\top \\ \hline \end{array}$$

is equivalent to

$$\begin{array}{|c|c|} \hline \mathbf{L}'_{22} & \mathbf{L}'_{22}^\top \\ \hline \end{array} = \begin{array}{|c|c|} \hline \mathbf{L}_{22} & \mathbf{L}_{22}^\top \\ \hline \end{array} + \begin{array}{|c|} \hline \mathbf{l}_i \\ \hline \end{array} \begin{array}{|c|} \hline \mathbf{l}_i^\top \\ \hline \end{array}.$$

2.5 The coordinate descent method

After having presented the two closely related active set and homotopy methods, it is useful to mention another one that qualifies as state-of-the-art at the time of writing this thesis. Contrary to the previous sections, no new contribution is brought here about this algorithm, and we rather just describe it with the same notations as the previous ones, in order to discuss, in the chapter 3, the respective merits of the three algorithms in various settings.

The first instance of this method appeared in (Fu, 1998), as the *shooting* algorithm, where it used as a starting point the unconstrained least-squares solution. Although unnecessary, this point was often regarded as discarding the algorithm for overdetermined systems or highly-regularized problems, and, together with the fact that it was not the main subject of the publication, prevented it to get the attention it deserved. It gained credit when presented in (Friedman et al., 2007) for a class of convex problems.

The general principle of coordinate descent is very simple and used in various communities on optimization problems. It consists in optimizing a function with vector input by repeatedly fixing all components but one as constants and optimizing on the single remaining variable. One typically cycles through the set of variables, but can also adapt the focus put on different variables, although an adaptive strategy can easily incur more cost than benefits. We present the basic cycling version, usually referred to as cyclic[all] coordinate descent (CCD). It can naturally only be applied, as is, on problems with a finite number of variables. However, we discuss in section 3.2.1.1, how it may be adapted to cope with infinite feature sets.

2.5.1 The one-dimensional LASSO

Let us first derive to what amounts solving the LASSO on a single feature ϕ .

Let β and β' be its coefficient, respectively before and after optimization, and \mathbf{r} and \mathbf{r}' the corresponding residuals. We have

$$\mathbf{r}' = \mathbf{r} + \beta\phi - \beta'\phi$$

where $\mathbf{r} + \beta\phi$ is the non-variable part of the residual.

The only two possible active sets are $\{\phi\}$ and \emptyset . Let us assume $\{\phi\}$ is optimal and compute the corresponding solution $\bar{\beta}$. If it is feasible we have the solution, otherwise $\beta' = 0$.

From section 2.1.2, we have

$$\begin{aligned} \beta' &= (\phi^\top \phi)^{-1} \left(\phi^\top (\mathbf{r} + \beta\phi) - \lambda \text{sign}(\beta') w(\phi) \right) \\ &= \frac{\phi^\top \mathbf{r} + \beta \|\phi\|_2^2 - \lambda \text{sign}(\beta') w(\phi)}{\|\phi\|_2^2} \\ &= \beta + \frac{\phi^\top \mathbf{r} - \lambda \text{sign}(\beta') w(\phi)}{\|\phi\|_2^2} \end{aligned}$$

if this equation has a solution, and $\beta' = 0$ otherwise (in case of sign disagreement). It is useful to normalize the features beforehand – and inversely scale their penalizations to keep the same problem, so that the expression simplifies. In this case we have

$$\beta' = \begin{cases} \beta + \boldsymbol{\phi}^\top \mathbf{r} - \lambda w(\boldsymbol{\phi}) & \text{if } \beta + \boldsymbol{\phi}^\top \mathbf{r} - \lambda w(\boldsymbol{\phi}) \geq 0 \\ \beta + \boldsymbol{\phi}^\top \mathbf{r} + \lambda w(\boldsymbol{\phi}) & \text{if } \beta + \boldsymbol{\phi}^\top \mathbf{r} + \lambda w(\boldsymbol{\phi}) \leq 0 \\ 0 & \text{otherwise} \end{cases}$$

or equivalently,

$$\beta' = \begin{cases} \beta + \boldsymbol{\phi}^\top \mathbf{r} - \lambda w(\boldsymbol{\phi}) & \text{if } \beta + \boldsymbol{\phi}^\top \mathbf{r} \geq \lambda w(\boldsymbol{\phi}) \\ \beta + \boldsymbol{\phi}^\top \mathbf{r} + \lambda w(\boldsymbol{\phi}) & \text{if } -\beta - \boldsymbol{\phi}^\top \mathbf{r} \geq \lambda w(\boldsymbol{\phi}) \\ 0 & \text{otherwise} \end{cases}$$

and equivalently, since $\lambda w(\boldsymbol{\phi})$ is positive,

$$\beta' = \begin{cases} \beta + \boldsymbol{\phi}^\top \mathbf{r} - \text{sign}(\beta + \boldsymbol{\phi}^\top \mathbf{r}) \lambda w(\boldsymbol{\phi}) & \text{if } |\beta + \boldsymbol{\phi}^\top \mathbf{r}| \geq \lambda w(\boldsymbol{\phi}) \\ 0 & \text{otherwise} \end{cases}$$

Repeating this one-dimensional optimization by cycling over the dictionary \mathcal{D} gives the algorithm 8.

2.6 Experiments

In order to estimate qualitative and quantitative properties of the three methods described in this chapter, we have run some experiments that illustrate the evolution of LASSO estimators for different values of the regularization parameter, the computational times, and the sequence of solutions computed by each method.

2.6.1 Illustration of LASSO estimators

We first illustrate the effect of ℓ^1 regularization on a problem where the observations are generated from a smooth function on $\mathcal{X} = \mathbb{R}$ and Gaussian white noise, for which the observations and estimators can be visualized.

The generative function is

$$f(x) = \frac{\sin(25x)}{\sqrt{6 - 5x}},$$

and the n observations are added identical independent Gaussian noise:

$$\forall i = 1, \dots, n, \quad y_i = f(x_i) + 0.1\nu, \quad \nu \sim \mathcal{N}(0, 1)$$

Algorithm 8 Cyclic Coordinate Descent for the regularized LASSO

Input: input vector $\mathbf{x} \in \mathcal{X}^n$, response vector $\mathbf{y} \in \mathbb{R}^n$, finite feature dictionary $\mathcal{D} \subset \mathbb{R}^{\mathcal{X}}$, penalization function $w : \mathcal{D} \rightarrow \mathbb{R}_+$, regularization parameter $\lambda \in \mathbb{R}_+$

Output: β converges to

$$\text{LASSO}_r(\mathbf{x}, \mathbf{y}, \mathcal{D}, w, \lambda) = \arg \min_{(\sigma, \beta) \in 2^{\mathcal{D}} \times \mathbb{R}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \beta\|_2^2 + \lambda \|\beta\|_{w_\sigma}$$

- 1: $\beta \leftarrow$ starting point.
- 2: $\mathbf{r} \leftarrow \mathbf{y} - \mathbf{X}^\top \beta$
- 3: **loop**
- 4: **for** $\phi \in \mathcal{D}$ **do**
- 5: $c \leftarrow \beta_\phi + \phi^\top \mathbf{r}$
- 6: $\beta' \leftarrow$ **if** $|c| > \lambda w(\phi)$ **then** $c - \text{sign}(c)\lambda w(\phi)$ **else** 0
- 7: **if** $\beta' \neq \beta_\phi$ **then**
- 8: $\mathbf{r} \leftarrow \mathbf{r} - (\beta' - \beta_\phi)\phi$
- 9: $\beta_\phi \leftarrow \beta'$
- 10: **end if**
- 11: **end for**
- 12: **end loop**

The features dictionary is composed by 10000 Gaussian functions with 100 possible centres (c) in $[0, 1]$ and 100 possible bandwidths σ in $[0, 1]$ also, these two parameters being taken on a uniform grid in $[0, 1]^2$:

$$\phi_{c,\sigma}(x) = \exp\left(-\left(\frac{x-c}{\sigma}\right)^2\right).$$

The common usage, followed here, is, for a fair application of the LASSO, to normalize the features on the observation set, so that they have equal ℓ^2 norms: given an arbitrary dictionary \mathcal{D} ,

$$\forall \phi \in \mathcal{D}, \quad \phi \leftarrow \frac{1}{\|\phi\|_2} \phi$$

or equivalently

$$\forall \phi \in \mathcal{D}, \quad w(\phi) = \|\phi\|_2.$$

This ensures that a feature is not selected just from its bigger scale and the consequent lower coefficient needed to achieve the same magnitude of effect. The fairness induced by this normalization can be understood by noting that considering, prior to the knowledge of the observations, a flat, null target $\mathbf{y} = \mathbf{0}$, the residual when assigning a coefficient w to any feature ϕ is then $\|\mathbf{y} - w\phi\|_2^2 = w\|\phi\|_2^2 = w$, hence coefficients of the same range have a priori similar effects on the residual for all features, and the selection is affected only by shape fitting. An other way of seeing this is that the correlation of a feature ϕ to the residual \mathbf{r} ,

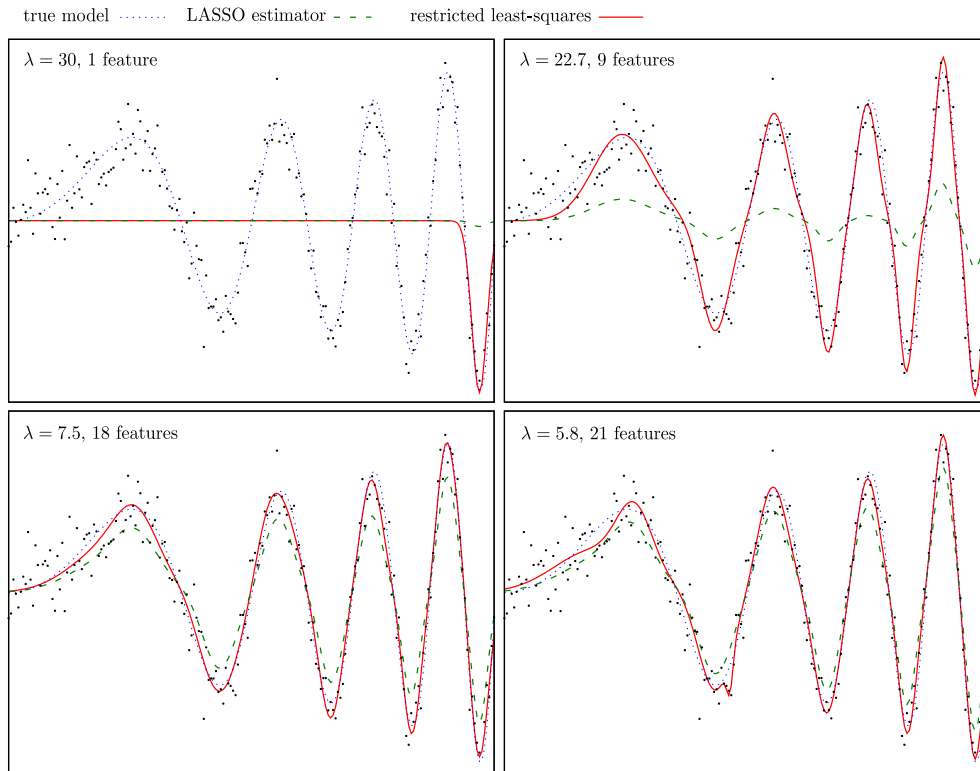


Figure 2.5: LASSO and restricted-least-squares (least-squares on active features) estimators for different values of λ , for noisy observations of a sinusoidal function, and 10000 Gaussian features.

The four estimators are, successively, the first one with one feature, the first to approximately fit all sinusoidal waves, the closest to the true model, and the first to be noticeably overfitting.

which is the selection criterion, geometrically corresponds to the projection of ϕ on \mathbf{r} , and equal norms of the features yields a selection of the feature that forms the least angle with \mathbf{r} , hence presents the most similar direction/shape with it.

In figure 2.5 are plotted the observations and the LASSO estimators for different values of λ , together with the corresponding restricted least-squares estimators, that is the least squares solutions when restricting to the selected features, i.e. the coefficient β^* used in the homotopy and ASD algorithms. It clearly shows that LASSO estimators should not be used as is, especially for large values of λ , since the ℓ^1 penalization not only has a selection property, but also limit the range of the coefficients, thus the ability to significantly reduce the residual. The restricted-least-squares solution is preferable, and is directly available in the homotopy and ASD algorithms. However, as can be seen in the figure, for a low value of λ that selects overfitting features (matching a single or a few points), the LASSO solution, although still not “reaching” the underlying model, also does not fully exploits the overfitting features, and may be preferable in this case.

2.6.2 Speed trials

In (Friedman et al., 2007), experiments were conducted to assess the better results of the CCD method in terms of speed, compared to the homotopy. We reproduced these experiments with our implementation of the homotopy method, and adding the new active set method for the regularized formulation.

The settings were the following:

- p features describing n observations were randomly generated from identical laws $\mathcal{N}(0, 1)$, such that the values of one feature are independent from each other, but the features have an expected population correlation ρ with one another;
- the response values were generated by assigning exponentially decreasing coefficients with alternating signs to these features, and adding Gaussian noise:

$$\mathbf{y} = \sum_{i=1}^p \beta_i \phi_i + \mathcal{N}(\mathbf{0}, kI)$$

where $\beta_i = (-1)^i \exp(-(j-1)/10)$, and k is chosen so that the signal to noise ratio equals 0.3.

The time was measured for the three algorithms to compute the LASSO regularization path. The homotopy computes by nature the precise path, and the other two were given a sequence of $\max(n, p)$ values for λ , using one solution as a warm start for computing the next one. Contrarily to the experiments run in (Friedman et al., 2007), all algorithms were here similarly implemented, in the same programming language (C) by the author, in the same framework, and the version of the homotopy that has been presented in this thesis uses a simplified formulation in terms of variables. This explains the different results obtained here: the absence of matrix operations in the CCD does not compensate the larger number of steps, in terms of computation speed, as was reported in the initial experiments. Indeed, CCD is slightly faster only for under-determined systems ($n > p$) and uncorrelated features, and significantly slower in other cases. The ASD algorithm used the same order of time as the homotopy, as could be expected, but consistently less. The difference can be more significant when using less steps (different values of λ) because, as observed in further section 2.6.3, ASD can be used to compute only a few actual LASSO solutions, and use the intermediate tentative estimators that descend to those as surrogates for the intermediate LASSO solutions.

2.6.3 LASSO path and descent paths

The homotopy algorithm computes the LASSO regularization path, that is, all LASSO solutions corresponding to meaningful values of λ ($\max_{\phi \in \mathcal{D}} \phi^T \mathbf{y}$ to 0). ASD and CCD converge to one LASSO solution, for a given value of λ , by

Table 2.1: Speed trial experiments with the same settings as in (Friedman et al., 2007).

The running times, in seconds, are averaged over 10 runs. All methods and trials were implemented in C in similar fashions.

n	p	Method	Population correlation between features					
			0	0.1	0.2	0.5	0.9	0.95
100	1000	homotopy	0.13	0.12	0.13	0.14	0.14	0.14
		ASD	0.09	0.09	0.09	0.10	0.10	0.10
		CCD	0.21	0.21	0.24	0.46	1.21	2.64
	5000	homotopy	0.63	0.65	0.62	0.67	0.62	0.60
		ASD	0.53	0.56	0.54	0.59	0.54	0.54
		CCD	1.39	1.39	1.54	2.32	7.39	8.53
	20000	homotopy	2.39	3.06	2.69	3.22	3.30	3.36
		ASD	2.06	2.56	2.25	2.68	2.76	2.85
		CCD	5.38	7.22	6.06	11.14	37.24	47.07
1000	100	homotopy	0.22	0.22	0.22	0.22	0.20	0.16
		ASD	0.19	0.19	0.19	0.19	0.17	0.13
		CCD	0.18	0.23	0.30	0.65	1.62	2.13
	5000	homotopy	0.71	0.70	0.70	0.70	0.70	1.05
		ASD	0.67	0.67	0.67	0.66	0.65	0.97
		CCD	0.61	0.75	0.91	1.44	4.31	8.07

following a descent path of estimators. In the absence of an a priori knowledge of a good value of λ , the latter two algorithms must thus be run successively for a selected number of different values, and can use one result as a warm start for the next run. It is interesting to characterize empirically the intermediate estimators that converge to each LASSO solution in this case. This was done on two problems: one with fully random features and responses, and the other using the sinusoidal model with Gaussian features as used in section 2.6.1.

When comparing the resulting LASSO path and descent paths, by plotting the squared residual against the coefficient's ℓ^1 -norm, as shown in figure 2.6, one can see that the CCD descent path is significantly different from the LASSO path: intermediate estimators immediately achieve a residual comparable to that of the LASSO, but with a high ℓ^1 -norm and large number of active features, and then the latter two are progressively decreased. The descent paths of ASD are much more similar to the LASSO path, by decreasing almost proportionally both the residual's ℓ^2 -norm and the coefficient's ℓ^1 -norm. This seems to indicate that, even when only a few true LASSO solutions are computed by this algorithm, the intermediate estimators built when converging to those may be considered as valid candidates: one can choose among them by, for example, monitoring the residual on a validation set, and then either use the selected one as is, as the regression estimator, or deduce from it a good ℓ^1 constraint before computing

the true corresponding LASSO estimator. In order to look further into this potential validity of ASD intermediate estimators, we also plotted, in addition to the LASSO elements that are the residual's ℓ^2 -norm and the coefficient's ℓ^1 -norm, the quantities that are the initial motivation to the LASSO: the number of active features (ℓ^0 constraint) against the least squared residual achieved with these selected features. As can be seen in the right part of figure 2.6, the results validate the intuition, since the intermediate estimators of ASD are even better than the LASSO ones with regard to this criterion: for an equal number of selected features, they consistently yield lower residuals than LASSO solutions. This is true even when directly running ASD for a single, very low value of the regularization parameter, which can raise the idea of a new algorithm, that is both very simple and somewhat peculiar, consisting in one run of ASD for a LASSO problem with a regularization parameter λ equal to zero. Thus no real LASSO solution is computed, but the descent path that leads to the (unconstrained) least-squares may yield, from empirical evidences, better estimators in terms of combined sparsity and low residuals.

2.6.3.1 The Non-Negative Forward Stepwise Algorithm

When fixing λ to zero, the resulting algorithm consists in repeatedly activating the most correlated feature and computing the – unregularized – least-squares estimators on the active features, just as done by the Forward Stepwise (FS) algorithm, but with the following additional step: when the sign of a feature's coefficient changes, it is inactivated. More precisely, the first feature of which the coefficient is zeroed when moving linearly from an estimator to the next is inactivated, and the least-squares recomputed on the reduced active set. It is thus a Forward Stepwise algorithm that also operates deselection of features on a somewhat natural basis: when a feature changes sign, it can be interpreted as not participating to the estimator in the same way it did when being selected, and it is natural to inactivate it.

However, a few experiments can show that such inactivations occur very rarely in practice, and the non-negative version of Forward Stepwise does not give significantly different results than the original version. The FS procedure *does* indeed produce itself better estimators than the LASSO in terms of ℓ^0 regularization, but ℓ^1 regularization is not simply a computable surrogate to ℓ^0 regularization. It also offers better generalization properties, because it limits both the number of active features and the range of their coefficients; that is why a LASSO estimator has a lower amplitude than the observations (see fig. 2.5), which makes it fit the general shape of the data rather than individual points. That is why the ℓ^0 penalty and FS have good selection properties but a limited regularization effect, i.e. a tend top over-fit the data quickly. Despite the quasi-equivalence to the regular FS algorithm and the poor generalization qualities, we give a sketch of the ASD algorithm in the limit case $\lambda = 0$ in Algorithm 9, under the name *Non-Negative Forward Selection*, for instructional purpose. “Non-Negative” accounts

for the fact that a feature is – greedily – selected, but only while its coefficient remains non-negative, in the positivity-trick view, or keeps its original sign, in the equivalent sign-monitoring view.

Algorithm 9 Non-Negative Forward Stepwise

Features are sequentially selected on the basis of their correlation to the restricted least-squares being the highest (classical Forward Stepwise procedure), and are also unselected when their coefficient is zeroed while going linearly from an estimator to the next one. It is equivalent to solving the normal, unregularized least-squares problem by ASD. In practice, deselections occur very rarely and do not significantly change the results from those of the regular Forward Stepwise algorithm.

Input: input vector $\mathbf{x} \in \mathcal{X}^n$, response vector $\mathbf{y} \in \mathbb{R}^n$, feature dictionary $\mathcal{D} \subset \mathbb{R}^{\mathcal{X}}$, penalization function $w : \mathcal{D} \rightarrow \mathbb{R}_+$.

Output: A sequence of linear estimators with decreasing residual's ℓ^2 -norm, that are the least-squares estimators on a globally increasing number of selected features.

Definition: For a feature function $\phi \in \mathcal{D}$, the bold notation denotes the application of ϕ to the input vector \mathbf{x} : $\boldsymbol{\phi} = \boldsymbol{\phi}(\mathbf{x}) = (\phi(x_1), \dots, \phi(x_n))^T$.

- 1: $\sigma \leftarrow \{\}$, $\mathbf{X} \leftarrow []$, $\boldsymbol{\beta} \leftarrow ()$
 - 2: **repeat**
 - 3: **repeat**
 - 4: $\bar{\boldsymbol{\beta}} \leftarrow (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{y}$
 - 5: $(\gamma, \phi) \leftarrow \min, \arg \min_{\phi \in \sigma} \left(\frac{\beta_\phi}{\beta_\phi - \bar{\beta}_\phi} \right)_{\geq 0}$
 - 6: **if** $\gamma \leq 1$ **then**
 - 7: $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \gamma(\bar{\boldsymbol{\beta}} - \boldsymbol{\beta})$
 - 8: $\sigma \leftarrow \sigma \setminus \{\phi\}$
 - 9: shrink $\mathbf{X}, \boldsymbol{\beta}$ accordingly
 - 10: **end if**
 - 11: **until** $\gamma \geq 1$
 - 12: $\boldsymbol{\beta} \leftarrow \bar{\boldsymbol{\beta}}$
 - 13: **output/consider** $(\sigma, \boldsymbol{\beta})$
 - 14: $\phi \leftarrow \arg \max_{\phi \in \mathcal{D}} |\boldsymbol{\phi}(\mathbf{y} - \mathbf{X}^T\boldsymbol{\beta})|/w(\phi)$
 - 15: $\sigma \leftarrow \sigma \cup \{\phi\}$
 - 16: extend \mathbf{X} by row $\boldsymbol{\phi}$, $\boldsymbol{\beta}$ by 0
 - 17: **until** $|\sigma| = n$ or satisfying estimator found
-

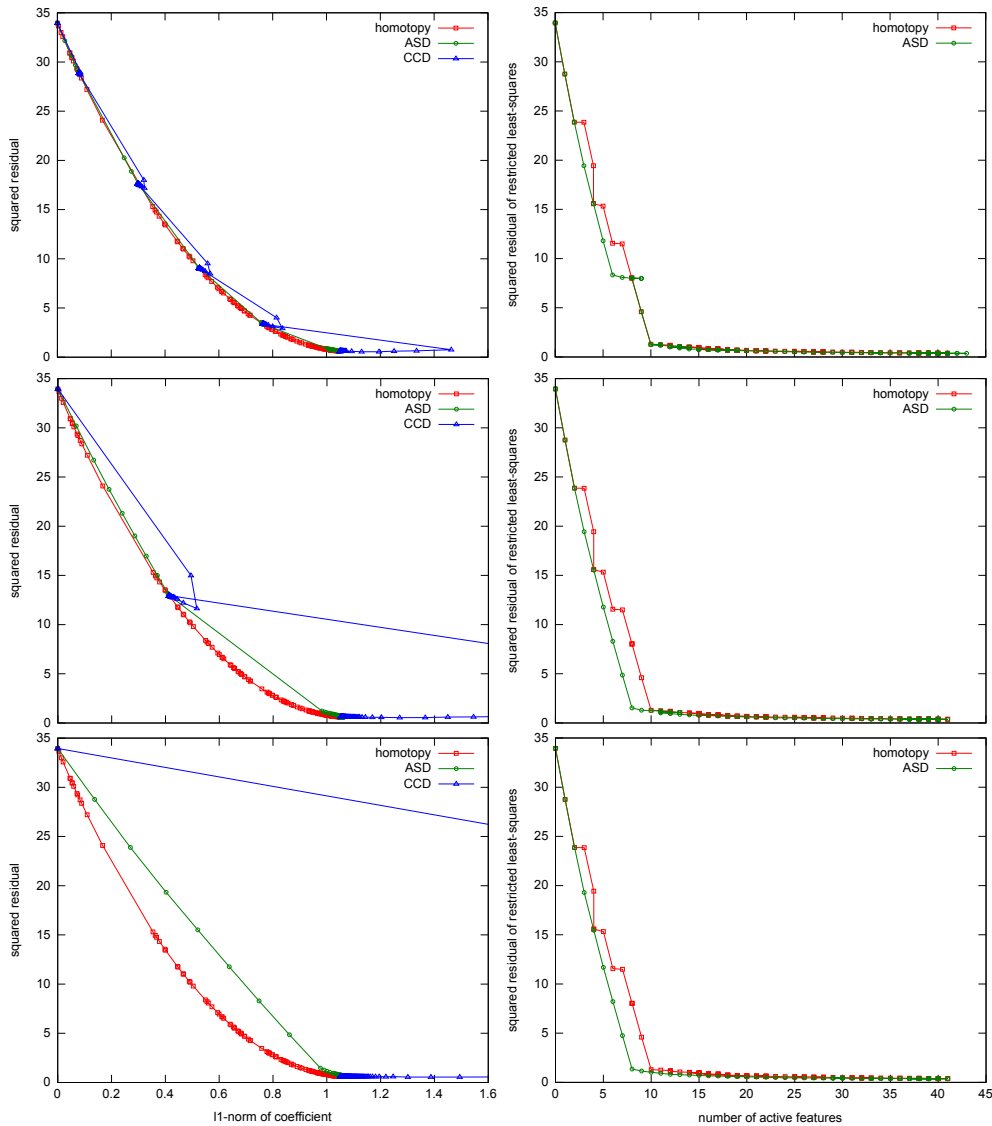


Figure 2.6: Characterization of the sequence of estimators generated by the homotopy, active set descent, and cyclical coordinate descent algorithms, on the sinusoidal problem illustrated in figure 2.5.

On the left are plotted the values minimized by the LASSO: the squared residual against the ℓ^1 -norm of the coefficient. The homotopy gives the LASSO solutions for every value of λ (from $\lambda_{\max} \approx 34$ to $0.05\lambda_{\max}$ here), while ASD and CCD compute k LASSO solutions and produce intermediate descent points that are LASSO-suboptimal ($k = 5, 2, 1$ from top to bottom). The intermediate points of ASD do not deviate much from LASSO optimality, contrary to the CCD points. On the right are plotted the feature-selection characteristics of the estimators of homotopy and ASD: the squared residual of the least-squares estimators on the selected features, against the number of selected features. The results show that the ASD intermediate steps are not only close to LASSO solutions, but even more relevant, in terms of feature selection. Figure 2.8 shows the bottom-right plot ($k = 1$) with a logarithmic scale, on which it can be seen that the restricted least-squares is consistently lower for ASD steps compared to LASSO solutions.

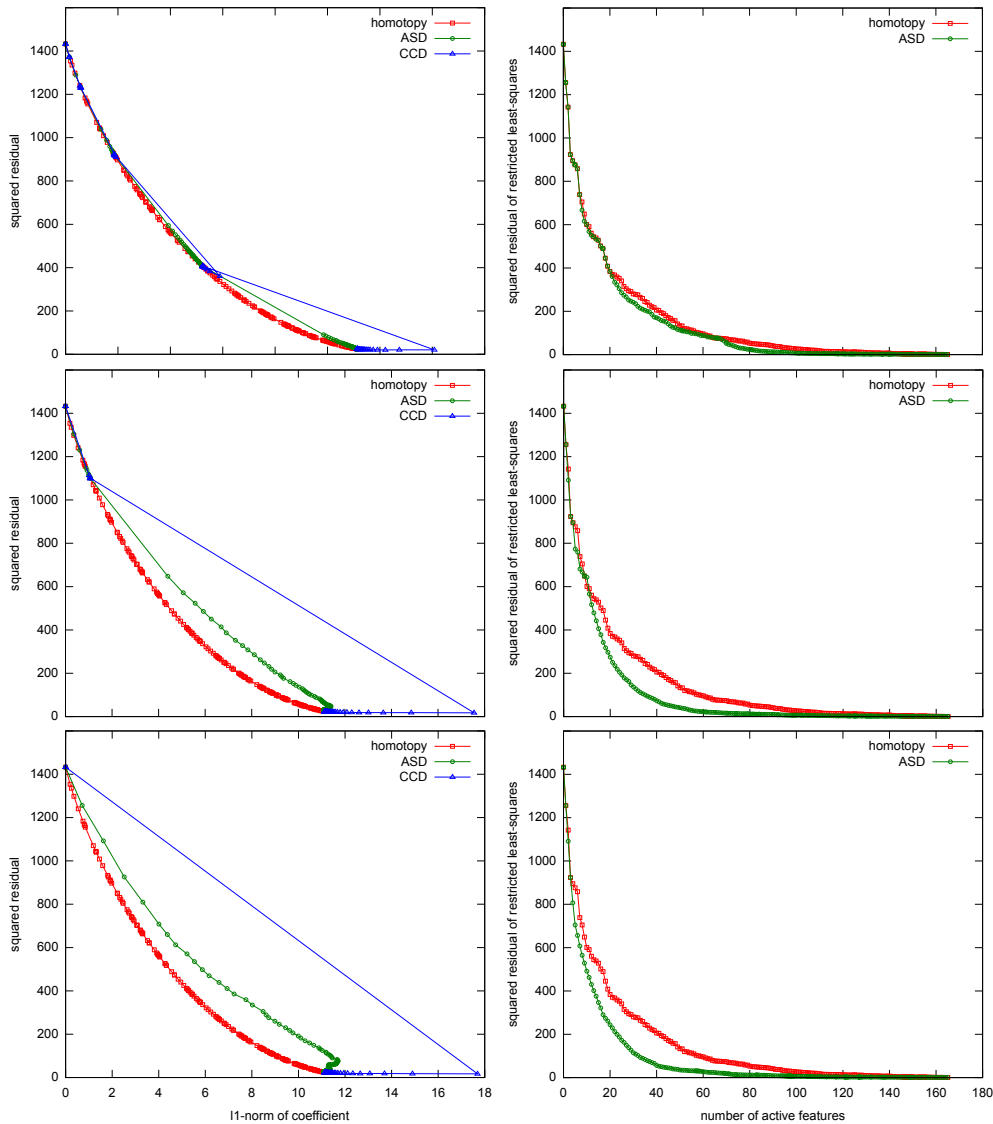


Figure 2.7: Characterization of the sequence of estimators generated by the homotopy, active set descent, and cyclical coordinate descent algorithms, on the speed trials problem of section 2.6.2, with $n = 200$ and $p = 10000$.

On the left are plotted the values minimized by the LASSO: the squared residual against the ℓ^1 -norm of the coefficient. The homotopy gives the LASSO solutions for every value of λ (from $\lambda_{\max} \approx 1430$ to $0.05\lambda_{\max}$ here), while ASD and CCD compute k LASSO solutions and produce intermediate descent points that are LASSO-suboptimal ($k = 5, 2, 1$ from top to bottom). The intermediate points of ASD do not deviate much from LASSO optimality, contrary to the CCD points. On the right are plotted the feature-selection characteristics of the estimators of homotopy and ASD: the squared residual of the least-squares estimators on the selected features, against the number of selected features. The results show that the ASD intermediate steps are not only close to LASSO solutions, but even more relevant, in terms of feature selection. This phenomenon is more pronounced on this artificial, randomized problem than on the more realistic sinusoidal problem of figure 2.6.

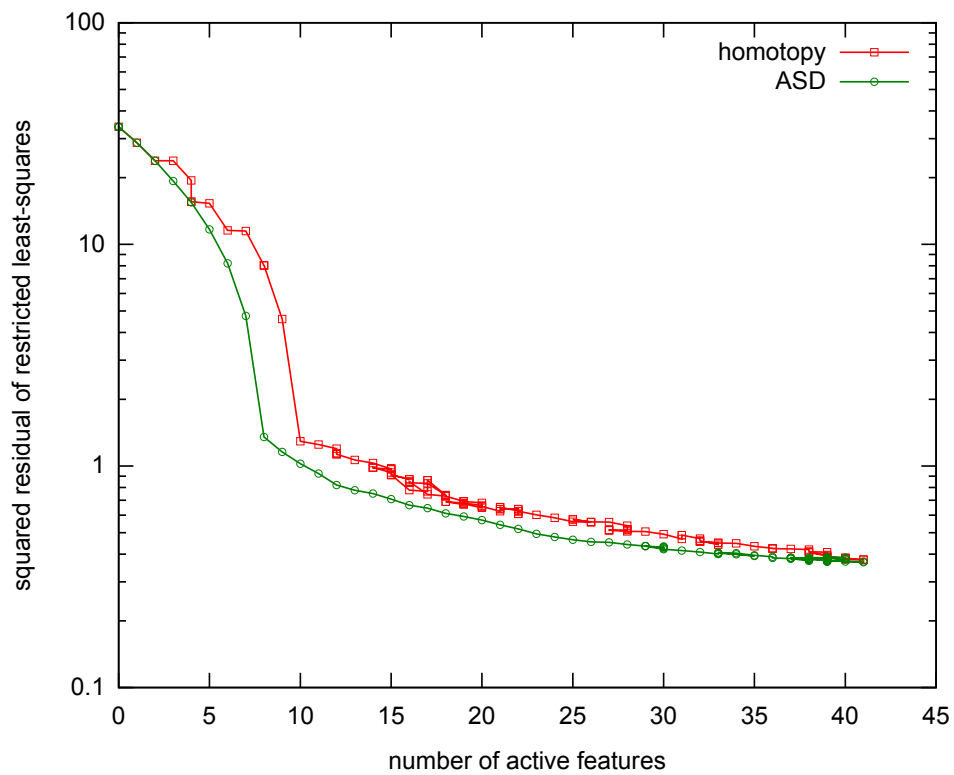


Figure 2.8: The bottom-right plot of figure 2.6 with logarithmic scale. The homotopy algorithm computes all LASSO solutions, while ASD descends to the LASSO solution for $\lambda = 0.05\lambda_{\max}$. However, the descent steps of ASD yields consistently better estimators in terms of features selection: the least-squares estimators on active features have lower residuals, for equal numbers of features.

Beyond the Simple LASSO

In chapter 2, the new derivation and adaptation of the active set descent to both forms of the LASSO, as well as the rewriting of the homotopy, were conducted, similarly to the previous expositions of these algorithms, under some regularity assumptions, and for classical settings.

By regularity, we mean we have assumed that no special case occurs that yields ambiguity in the choices of features to activate/inactivate, multiplicity of solutions, or failure of some computations, like the inversion of the Gram matrix. Such degenerating cases are discussed in section 3.1.

Settings are classical in several ways. Although we have given a feature-oriented definition of the LASSO that handles inputs from an arbitrary set and predictors in the form of an arbitrary set of features, we have not yet analyzed to which extent, and at the price of which tailoring, these algorithms can handle the particular setting of an infinite feature set. This is the object of section 3.2.

The data were also considered to be given and treated in a *batch* fashion, that is they are all given at once and the solution is computed by treating them as a whole. Several situations require, or benefit from treating the data in sequence, whether through *online learning* where the data is received and treated as a stream, or more generally *sequential learning*, in which the experimenter may also face a sequence of similar or related problems. These settings are addressed in section 3.3.

Finally, we focused on the LASSO, given both the simplicity and frequent relevance of the squared-residual loss, and the practical aspect and relevance of the ℓ^1 regularization for inducing/recovering sparsity. However, alternative loss functions are more and more commonly used, as well as extensions of the simple ℓ^1 regularization, that take into account specific properties of the data or induce additional properties of the estimator. Some of these alternatives are considered in section 3.4.

In this chapter is studied to what extent, and in which manner, the three algorithms given in chapter 2 (ASD, homotopy, CCD, all three for the regularized formulation) can handle and adapt to these situations and alternative problems.

For these discussions and analyses of the algorithms, it is simpler, rather than considering a penalization factor $w(\phi)$ for each feature ϕ , to scale them by this factor, with the following substitution:

$$\mathcal{D}' = \left\{ \frac{\phi}{w(\phi)} \mid \phi \in \mathcal{D} \right\}$$

Thus, given an estimator defined with the original features:

$$\hat{f}(\mathbf{x}) = \sum_{\phi \in \mathcal{D}} \beta_{\phi} \phi(\mathbf{x}),$$

its equivalent with the scaled features is

$$\hat{f}(\mathbf{x}) = \sum_{\phi \in \mathcal{D}} \underbrace{w(\phi)\beta_{\phi}}_{\beta'_{\phi}} \frac{\phi(\mathbf{x})}{w(\phi)},$$

and the – simple unweighted – ℓ^1 -norm of the new coefficient β' is

$$\|\beta'\|_1 = \sum_{\phi \in \mathcal{D}'} |w(\phi)\beta_{\phi}| = \sum_{\phi \in \mathcal{D}'} w(\phi)|\beta_{\phi}| = \|\beta\|_w$$

Hence, a LASSO problem with penalization factors can be transformed into an equivalent classical LASSO problem using unweighted ℓ^1 -norm. Although penalization factors are more convenient than feature scaling for the implementation and to understand their effects, we may switch to the simple ℓ^1 -norm view to clarify the explanations. One of the consequences is that the optimality conditions (zeroing of the gradient for active features, and its positivity for inactive features), translates into the correlation of a feature with the residual being equal, respectively greater than λ , rather than $\lambda w(\phi)$, and we may designate the three possible states of a feature with respect to a current estimator as the following:

$$\begin{cases} |\phi^{\top} \mathbf{r}| < \lambda & \text{under-correlation,} \\ |\phi^{\top} \mathbf{r}| = \lambda & \text{equi-correlation,} \\ |\phi^{\top} \mathbf{r}| > \lambda & \text{over-correlation.} \end{cases}$$

3.1 Degeneracies

A *degeneracy* or *degenerate case* is a general term for referring to a situation that has a probability zero of happening under “normal” circumstances, and is frequently assumed not to happen. For example, if \mathbf{X} is an $n \times p$ matrix with $n \geq p$, the assumption that $\mathbf{X}\mathbf{X}^{\top}$ is invertible is not true if \mathbf{X} is not full rank, that is, for example, one of its p rows is a linear combination of some other rows. If the entries of \mathbf{X} are drawn randomly following, for example, independent identical uniform distributions over some interval, this event has a probability zero; however, artificial data may cause such situation, and, of course, the probability zero is a theoretical concept and becomes nonzero with the implementation on a computer, that uses discrete storage of numbers.

The two sources of degeneracies in the LASSO are:

- the existence of multiple solutions,
- the simultaneous violations of constraints in the homotopy method.

These two problems were not thoroughly studied in existing literature, or tackled by mentioning their low likeliness and/or “brute force” detection and workarounds. However, they can arise in practise, and simpler and more integrated treatments can be derived, as we show in the following, where we give a closer look at these cases.

3.1.1 Multiple solutions

In order to analyse the conditions and characteristics of multiple solutions, let us start by recalling the existence and continuity of the LASSO path. From this fact, the existence of two distinct solutions implies that the path splits at some point, into two branches. We can discard the case where one of the branches corresponds to a break point (activation or inactivation of a feature) whereas the other branch incurs no change in the active set. Indeed, the coefficients and correlations evolve monotonically (linearly) on the path (i.e. wrt. λ), and (in)activations can and *must* occur at the points where they reach their bounds as defined by the constraints: inactivation when a feature’s coefficient reaches zero, and activation when a feature’s correlation reaches $\pm\lambda$; not proceeding them necessarily violates a constraint, and anticipating them (e.g. activation before equi-correlation) is necessarily suboptimal. Hence there can be an alternative only between two activations/inactivations, one “covering” for the other, that is, proceeding to any of them prevents the other’s violation to occur. Hence, the possibility of multiple solutions occurs at such a break point, and when several features reach their bound for status change at the exact same point in the path.

Before characterizing alternative choices at a break point, let us state a general result on distinct solutions. As pointed out in (Osborne et al., 2000b) (theorem 1), convexity arguments lead to the fact that if two distinct solutions exist, any of their convex combinations is also a solution. We establish the following theorem that gives a stronger statement:

Theorem 3.1.1. *If (σ_1, β_1) and (σ_2, β_2) are two solutions to a LASSO problem, the corresponding estimators are equal on all observed inputs.*

Proof. Let us first point out that features that are active in both estimators necessarily share the same sign for their coefficients, as this sign must agree with that of the features’ correlation to the residual.

Let $\sigma = \sigma_1 \cup \sigma_2$ be the smallest sufficient feature dictionary to define the two solutions, and let us switch from the sparse notation (σ_i, β_i) to the more classical one where β_1 and β_2 gather the coefficients on the whole dictionary σ , including zeroes. Let \mathbf{r}_1 and \mathbf{r}_2 be the residual vectors corresponding to the two estimators, defined by

$$\mathbf{r}_i = \mathbf{y} - \mathbf{X}_\sigma^\top \beta_i, \quad i \in \{1, 2\}$$

Let $\rho \in (0, 1)$ be a mixing coefficient, and $\bar{\rho} = 1 - \rho$.

β_1 , β_2 , and $\rho\beta_1 + \bar{\rho}\beta_2$ are all minimizers of the LASSO objective function, which implies there exists C such that

$$\|\mathbf{r}_1\|_2^2 + \lambda\|\beta_1\|_1 = C \quad (3.1)$$

$$\|\mathbf{r}_2\|_2^2 + \lambda\|\beta_2\|_1 = C \quad (3.2)$$

$$\|\mathbf{y} - \rho\mathbf{X}_\sigma^\top\beta_i - \bar{\rho}\mathbf{X}_\sigma^\top\beta_i\|_2^2 + \lambda\|\rho\beta_1 + \bar{\rho}\beta_2\|_1 = C. \quad (3.3)$$

Since the coefficients of a feature share the same sign in both estimators – and their affine combinations, and letting \mathbf{s} be the vector of these signs,

$$\|\rho\beta_1 + \bar{\rho}\beta_2\|_1 = (\rho\beta_1 + \bar{\rho}\beta_2)^\top \mathbf{s} \quad (3.4)$$

$$= \rho\beta_1^\top \mathbf{s} + \bar{\rho}\beta_2^\top \mathbf{s} \quad (3.5)$$

$$= \|\rho\beta_1\|_1 + \|\bar{\rho}\beta_2\|_1, \quad (3.6)$$

and (3.3) is equivalent to

$$\|\mathbf{y} - \rho\mathbf{X}_\sigma^\top\beta_i - \bar{\rho}\mathbf{X}_\sigma^\top\beta_i\|_2^2 + \rho\lambda\|\beta_1\|_1 + \bar{\rho}\lambda\|\beta_2\|_1 = C. \quad (3.7)$$

ρ and $\bar{\rho}$ summing to one, the first term can be rewritten so as to obtain

$$\|\rho\mathbf{r}_1 + \bar{\rho}\mathbf{r}_2\|_2^2 + \rho\lambda\|\beta_1\|_1 + \bar{\rho}\lambda\|\beta_2\|_1 = C. \quad (3.8)$$

Developing the first term gives

$$\rho^2\|\mathbf{r}_1\|_2^2 + \bar{\rho}^2\|\mathbf{r}_2\|_2^2 + 2\rho\bar{\rho}\mathbf{r}_1^\top\mathbf{r}_2 + \rho\lambda\|\beta_1\|_1 + \bar{\rho}\lambda\|\beta_2\|_1 = C. \quad (3.9)$$

The equality $\rho\bar{\rho} = \rho - \rho^2 = \bar{\rho} - \bar{\rho}^2$ allows to rewrite to

$$\rho^2\|\mathbf{r}_1\|_2^2 + \bar{\rho}^2\|\mathbf{r}_2\|_2^2 + 2\rho\bar{\rho}\mathbf{r}_1^\top\mathbf{r}_2 + \rho^2\lambda\|\beta_1\|_1 + \bar{\rho}^2\lambda\|\beta_2\|_1 + \rho\bar{\rho}\lambda\|\beta_1\|_1 + \rho\bar{\rho}\lambda\|\beta_2\|_1 = C, \quad (3.10)$$

and

$$\rho^2(\|\mathbf{r}_1\|_2^2 + \lambda\|\beta_1\|_1) + \bar{\rho}^2(\|\mathbf{r}_2\|_2^2 + \lambda\|\beta_2\|_1) + 2\rho\bar{\rho}\mathbf{r}_1^\top\mathbf{r}_2 + \rho\bar{\rho}\lambda\|\beta_1\|_1 + \rho\bar{\rho}\lambda\|\beta_2\|_1 = C, \quad (3.11)$$

i.e.

$$\rho^2C + \bar{\rho}^2C + 2\rho\bar{\rho}\mathbf{r}_1^\top\mathbf{r}_2 + \rho\bar{\rho}(\lambda\|\beta_1\|_1 + \lambda\|\beta_2\|_1) = C. \quad (3.12)$$

Substituting for $\lambda\|\beta_2\|_1$ and $\lambda\|\beta_1\|_1$ from (3.1) and (3.2) gives

$$\rho^2C + \bar{\rho}^2C + 2\rho\bar{\rho}\mathbf{r}_1^\top\mathbf{r}_2 + \rho\bar{\rho}(C - \|\mathbf{r}_1\|_2^2 + C - \|\mathbf{r}_2\|_2^2) = C, \quad (3.13)$$

and

$$(\rho^2 + \bar{\rho}^2 + 2\rho\bar{\rho})C + 2\rho\bar{\rho}\mathbf{r}_1^\top\mathbf{r}_2 - \rho\bar{\rho}(\|\mathbf{r}_1\|_2^2 + \|\mathbf{r}_2\|_2^2) = C, \quad (3.14)$$

$$C - \rho\bar{\rho}(\|\mathbf{r}_1\|_2^2 + \|\mathbf{r}_2\|_2^2 - \mathbf{r}_1^\top \mathbf{r}_2) = C, \quad (3.15)$$

and finally

$$\|\mathbf{r}_1 - \mathbf{r}_2\|_2^2 = 0, \quad (3.16)$$

which implies

$$\mathbf{r}_1 = \mathbf{r}_2 \quad (3.17)$$

The equality of residuals naturally implies equality of estimations at observed points:

$$\mathbf{X}_\sigma^\top \boldsymbol{\beta}_1 = \mathbf{X}_\sigma^\top \boldsymbol{\beta}_2 \quad (3.18)$$

□

The following corollary implies that even if some features in the dictionary are linear combinations of one another (meaning matrix $\mathbf{X}_\mathcal{D}$ has low rank for a finite dictionary), a LASSO problem has a unique solution if none of these combinations are affine:

Corollary 3.1.2. *Let λ be a value of the regularization parameter for which a unique solution \hat{f} of the LASSO uses k features $\sigma = \{\phi_1, \dots, \phi_k\}$, and beyond which (parameter $\lambda + \gamma$) a new feature must be activated.*

Let \hat{f}_a and \hat{f}_b be two candidate solutions, activating two distinct features ϕ_a and ϕ_b :

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^k \beta_i \phi_i \quad (3.19)$$

$$\hat{f}_a(\mathbf{x}) = \sum_{i=1}^k \beta_i \phi_i + \gamma \left(\sum_{i=1}^k \beta_i^a \phi_i + \beta_a \phi_a \right) \quad (3.20)$$

$$\hat{f}_b(\mathbf{x}) = \sum_{i=1}^k \beta_i \phi_i + \gamma \left(\sum_{i=1}^k \beta_i^b \phi_i + \beta_b \phi_b \right) \quad (3.21)$$

\hat{f}_a and \hat{f}_b are both solution to the LASSO with parameter $\lambda + \gamma$ only if ϕ_a (resp. ϕ_b), is an affine combination of the previously active features and ϕ_b (resp. ϕ_a) or their opposite :

$$\begin{aligned} \exists \alpha_1^a, \dots, \alpha_k^a, \alpha_b^a \in \mathbb{R}, \exists s_1^a, \dots, s_k^a, s_b^a \in \{-1, +1\}, & \left\{ \begin{array}{l} \phi_a = \sum_{i=1}^k \alpha_i^a (s_i^a \phi_i) + \alpha_b^a (s_b^a \phi_b) \\ \sum_{i=1}^k \alpha_i^a + \alpha_b^a = 1 \end{array} \right. \\ \exists \alpha_1^b, \dots, \alpha_k^b, \alpha_a^b \in \mathbb{R}, \exists s_1^b, \dots, s_k^b, s_a^b \in \{-1, +1\}, & \left\{ \begin{array}{l} \phi_b = \sum_{i=1}^k \alpha_i^b (s_i^b \phi_i) + \alpha_a^b (s_a^b \phi_a) \\ \sum_{i=1}^k \alpha_i^b + \alpha_a^b = 1 \end{array} \right. \end{aligned}$$

Proof. Let us suppose \hat{f}_a and \hat{f}_b are both solutions.

ϕ_a being a linear combination (on the observed points) of features from $\sigma \cup \{\phi_b\}$ is a direct consequence of the equality of the estimators that results from theorem 3.1.1:

$$\begin{aligned} \sum_{i=1}^k \beta_i \phi_i + \gamma \left(\sum_{i=1}^k \beta_i^a \phi_i + \beta_a \phi_a \right) &= \sum_{i=1}^k \beta_i \phi_i + \gamma \left(\sum_{i=1}^k \beta_i^b \phi_i + \beta_b \phi_b \right) \\ \sum_{i=1}^k \beta_i^a \phi_i + \beta_a \phi_a &= \sum_{i=1}^k \beta_i^b \phi_i + \beta_b \phi_b \\ \phi_a &= \sum_{i=1}^k \frac{\beta_i^b - \beta_i^a}{\beta_a} \phi_i + \frac{\beta_b}{\beta_a} \phi_b \end{aligned}$$

Let $\Delta\beta = \beta^b - \beta^a$, thus noting

$$\phi_a = \sum_{i=1}^k \frac{\Delta\beta_i}{\beta_a} \phi_i + \frac{\beta_b}{\beta_a} \phi_b. \quad (3.22)$$

Equivalently,

$$\phi_b = \sum_{i=1}^k \frac{-\Delta\beta_i}{\beta_b} \phi_i + \frac{\beta_a}{\beta_b} \phi_a. \quad (3.23)$$

The equality of residuals that results from theorem 3.1.1 implies that \hat{f}_a and \hat{f}_b must also have equal ℓ^1 -norms of coefficients, in order to achieve equal LASSO losses:

$$\sum_{i=1}^k |\beta_i + \gamma\beta_i^a| + \gamma|\beta_a| = \sum_{i=1}^k |\beta_i + \gamma\beta_i^b| + \gamma|\beta_b| \quad (3.24)$$

The interval that we consider for the regularization parameter ($\lambda + \gamma \in [\lambda, \lambda']$) is such that the active set does not change, which implies in particular that the linear coefficients in the solutions do not change signs :

$$\forall \gamma \in [0, \lambda' - \lambda], \forall i \in \{1, k\}, \quad \text{sign}(\beta_i) = \text{sign}(\beta_i + \gamma\beta_i^a) = \text{sign}(\beta_i + \gamma\beta_i^b)$$

which implies that the absolute value of all coefficients of $(\phi_i)_{i \in \{1, k\}}$ can be written as

$$\begin{aligned} |\beta_i + \gamma\beta_i^a| &= (\beta_i + \gamma\beta_i^a) \text{sign}(\beta_i) \\ &= \beta_i \text{sign}(\beta_i) + \gamma\beta_i^a \text{sign}(\beta_i) \\ &= |\beta_i| + \gamma\beta_i^a \text{sign}(\beta_i) \end{aligned}$$

and

$$\begin{aligned} |\beta_i + \gamma\beta_i^b| &= (\beta_i + \gamma\beta_i^b) \text{sign}(\beta_i) \\ &= \beta_i \text{sign}(\beta_i) + \gamma\beta_i^b \text{sign}(\beta_i) \\ &= |\beta_i| + \gamma\beta_i^b \text{sign}(\beta_i) \end{aligned}$$

Hence equation (3.24) is equivalent to

$$\sum_{i=1}^k |\beta_i| + \gamma\beta_i^a \text{sign}(\beta_i) + \gamma|\beta_a| = \sum_{i=1}^k |\beta_i| + \gamma\beta_i^b \text{sign}(\beta_i) + \gamma|\beta_b|$$

and

$$\sum_{i=1}^k \beta_i^a \text{sign}(\beta_i) + |\beta_a| = \sum_{i=1}^k \beta_i^b \text{sign}(\beta_i) + |\beta_b|$$

i.e.

$$|\beta_a| = \sum_{i=1}^k \Delta\beta_i \text{sign}(\beta_i) + |\beta_b|$$

This necessary condition for \hat{f}_a and \hat{f}_b to be both solutions also implies the affinities of (3.22) and (3.23):

$$\sum_{i=1}^k \frac{\Delta\beta_i}{|\beta_a|} \text{sign}(\beta_i) + \frac{|\beta_b|}{|\beta_a|} = 1$$

i.e.

$$\sum_{i=1}^k \frac{\Delta\beta_i}{\beta_a} \text{sign}(\beta_a\beta_i) + \frac{\beta_b}{\beta_a} \text{sign}(\beta_a\beta_b) = 1$$

in addition to (3.22) that can be rewritten as

$$\phi_a = \sum_{i=1}^k \frac{\Delta\beta_i}{\beta_a} \text{sign}(\beta_a\beta_i) \text{sign}(\beta_i) \phi_i + \frac{\beta_b}{\beta_a} \text{sign}(\beta_a\beta_b) \text{sign}(\beta_a\beta_b) \phi_b$$

and conversely for ϕ_b vis-à-vis ϕ_a .

□

We have thus shown that, even if the number of features is greater than the number of observations, or more generally some features are linearly dependent regarding their values on the observations, the ℓ^1 penalty gives a second criterion that distinguishes between them, yielding a unique LASSO solution, unless the linear dependence is affine, which means that assigning the same coefficient to one of these features or to the equivalent combination of the others incurs the

same cost in the coefficient's ℓ^1 norm and also the same decrease in the residual. However, this also means that the two options are really equivalent, and the choice can be made arbitrarily. This means no change for the Active Set method, that activates any over-correlated feature, and activating any of a group of affinely dependent ones makes them all equi-correlated; thus once an arbitrary choice is made, the problem is solved because the other options are discarded. Greater caution is needed for the homotopy method. In the formulations of (Osborne et al., 2000a) and (Efron et al., 2004), each step consists in computing the smallest step size in decreasing λ such that an inactive features becomes equi-correlated or an active feature is zeroed, and the equi-correlation point is computed for both possible signs of activation. If, taking the simplest example, a feature appears twice in the dictionary, once one of its occurrence is activated, the other one follows its clone, that is they both remain equi-correlated as long as the first one is active. This means that the second one will constantly candidate for immediate activation, although it should not. In our formulation, the next value of λ is computed rather than the change in its value. The resulting formula gives a value of zero, meaning no activation is required before the end of the regularization path, for all those "passively equi-correlated" features.

We have thus shown that when several features are zeroed or reach equi-correlation at the same point of the regularization path, the possibility that this gives way to distinct alternative solutions is unlikely, i.e. a degenerate case even if $|\mathcal{D}| > n$, and these solutions are equivalent. One can arbitrarily choose between them, and the algorithms as stated in the previous chapter need not be changed, since the unchosen features become and remain equi-correlated without being activated, and both algorithms track *over-correlation* rather than *equi-correlation*.

The other possibilities when two features reach equi-correlation at the same point are :

- the activation of one of them is sufficient because the incurred change of direction prevents the other from becoming over-correlated,
- the activation of both is necessary.

More generally, when an arbitrary number of equi-correlations/zeroings occur simultaneously, the problem arises of finding which subset of activations/inactivations emerges beyond this point.

3.1.2 Simultaneous status changes

This issue does not exist for the active set method, since it operates for a given value of λ , independently of what happens at different values. If λ presents a simultaneous-status-change singularity, this is not a problem in itself: all features that are equi-correlated *and* have a zero coefficient can be considered either active or inactive –the second option being preferred–, regardless of sign disagreements

or over-correlations that may occur beyond λ depending on their status. The problem exists for the homotopy method, which is based on computing when and how the solution changes with λ , and the “how” part is affected.

One can conjecture that no special care is actually needed to address the problem, i.e. one could follow the algorithm with arbitrary choices of features to activate/inactivate, followed by steps of size zero for λ . This is indeed the case when two features reach equi-correlation at once:

- if both features are necessary beyond, this will appear by simply activating any of them, compute the new regularization direction, along which the other feature necessarily becomes over-correlated, which results in its activation in the next step, and the optimal activation set being reached (both features activated), the algorithm goes its way beyond λ with no further complication;
- if one “dominates” the other, i.e. activating it extinguishes the need for the other, which translates by the correlation of the latter decreasing again, it is either activated first in which case the optimal activation set is reached, or the dominated one is activated first, which necessarily results in the dominant one’s over-correlation and activation, and the new direction must then give the wrong sign to the dominated feature, resulting in its inactivation; the last fact is ensured because by definition/construction, an unnecessary feature gets a negative (or non-compliant) sign in the formula $\beta = \beta^* - \lambda\Delta\beta$, as explained in the derivation of the active set descent method: a feature is unnecessary (inactive in the LASSO solution) if and only if the loss is lessened by a negative coefficient, i.e. by violating the positivity constraint.

However, it is not sure, and difficult to establish, if this generalizes to a larger number of simultaneous equi-correlations. For example, the following cycle may occur with four features $\phi_1, \phi_2, \phi_3, \phi_4$:

$$\begin{aligned}
 \phi_1 &\rightarrow \text{activate } \phi_2 \\
 \phi_1, \phi_2 &\rightarrow \text{inactivate } \phi_1 \\
 \phi_2 &\rightarrow \text{activate } \phi_3 \\
 \phi_2, \phi_3 &\rightarrow \text{inactivate } \phi_2 \\
 \phi_3 &\rightarrow \text{activate } \phi_4 \\
 \phi_3, \phi_4 &\rightarrow \text{inactivate } \phi_3 \\
 \phi_4 &\rightarrow \text{activate } \phi_1 \\
 \phi_4, \phi_1 &\rightarrow \text{inactivate } \phi_4 \\
 \dots &\rightarrow
 \end{aligned}$$

and if the optimal activation set is $\{\phi_1, \phi_3\}$, it is never reached. In the lack of a proof that such a cycle cannot happen, strategies that provably do not cycle must be considered.

Let us first note that the problem involves only the set \mathcal{C} of features that are equi-correlated and have a zero coefficient, whether they were active for greater values of λ and are zeroed at λ , or were inactive and become equi-correlated. The other – active or inactive – features can be left aside, since we are looking for the active set in the immediate neighbourhood of λ , that can be considered sufficiently close so that no other feature may be zeroed or over-correlated.

Hence, a first possibility is to identify the whole set \mathcal{C} , and do an exhaustive search over its subsets for the one that incurs sign-compliant coefficients for its members and no over-correlation for the unselected ones. This seems a reasonable solution, since it is extremely unlikely that \mathcal{C} is big. However, this may still occur with an ill-defined dataset, and is of combinatorial complexity; moreover, this method is a bit tedious to incorporate in the algorithm.

In remark 8 of (Osborne et al., 2000a), it is suggested that if such a cycle is detected, one can get past this singular point of the regularization path by setting t (resp. λ) to a slightly greater (resp. lower) value, then switch to the descent method for solving for that new point, and then switch back to homotopy when “back on track”. In (Efron et al., 2004), the solution of adding a jitter to the \mathbf{y} values is proposed. More generally, small random perturbations of the problem data can break the singularity without sensibly affecting its solution. Such a perturbation is also one of the possible workarounds for the similar cycling problem that occurs in linear programming. An elegant and efficient *exact* method was derived from it in (Wolfe, 1963), that uses *symbolic* perturbations that need not be quantified. Let us follow this approach of symbolically perturbing the data, and hopefully even find that the symbolic expression of the perturbation term can be dropped and simply leave a well-formed rule for selecting among simultaneous candidates for activation.

3.1.2.1 Breaking the tie

Rather than perturbing the response \mathbf{y} , one can change the penalizations of the features, or equivalently scale these features. This involves less entities, that are also more related to the problem: $|\mathcal{C}|$ features rather than n observations. A slight increase in the penalization of all candidate features delays their equi-correlations. A single scale of the penalizations by a common factor $\gamma > 1$ has a probability one to break the tie: the evolution of correlations with λ is linear, the scale of penalizations results in a scale of the slopes, and if they are different for each feature, the tie is broken. Let us however protect against a new singularity (equal slopes) by allowing different scales for each feature: we scale the penalization of each candidate feature ϕ by $(1 + \varepsilon_\phi)$.

For all features ϕ of \mathcal{C} , let us increase the penalization by scaling θ_ϕ (their signed penalization) by $(1 + \varepsilon_\phi) > 1$, and get

$$\phi^\top(\mathbf{r}^* + \lambda\Delta_{\mathbf{r}}) < \lambda((1 + \varepsilon_\phi)\theta_\phi)$$

Equi-correlation is thus delayed to a further point, that is a lower value of λ . Let

$\lambda - \delta_\phi$ be this point. We are looking for the closest one, that is the least δ_ϕ . δ_ϕ is given by

$$\begin{aligned}\phi^\top(\mathbf{r}^* + (\lambda - \delta_\phi)\mathbf{\Delta}_r) &= (\lambda - \delta_\phi)(1 + \varepsilon_\phi)\theta_\phi \\ \phi^\top(\mathbf{r}^* + \lambda\mathbf{\Delta}_r) - \delta_\phi\phi^\top\mathbf{\Delta}_r &= \lambda\theta_\phi + \varepsilon_\phi\lambda\theta_\phi - \delta_\phi(1 + \varepsilon_\phi)\theta_\phi \\ -\delta_\phi\phi^\top\mathbf{\Delta}_r &= \varepsilon_\phi\lambda\theta_\phi - \delta_\phi(1 + \varepsilon_\phi)\theta_\phi \\ \delta_\phi((1 + \varepsilon_\phi)\theta_\phi - \phi^\top\mathbf{\Delta}_r) &= \varepsilon_\phi\lambda\theta_\phi \\ \delta_\phi &= \frac{\varepsilon_\phi\lambda}{1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi} + \varepsilon_\phi}\end{aligned}$$

Let us see to what amounts minimizing this value over \mathcal{C} :

$$\begin{aligned}\arg \min_{\phi \in \mathcal{C}} \delta_\phi &= \arg \min_{\phi \in \mathcal{C}} \frac{\varepsilon_\phi\lambda}{1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi} + \varepsilon_\phi} \\ &= \arg \min_{\phi \in \mathcal{C}} \frac{\varepsilon_\phi}{1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi} + \varepsilon_\phi} \\ &= \arg \max_{\phi \in \mathcal{C}} \frac{1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi} + \varepsilon_\phi}{\varepsilon_\phi} \\ &= \arg \max_{\phi \in \mathcal{C}} \frac{1}{\varepsilon_\phi} \left(1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi} \right) + 1 \\ &= \arg \max_{\phi \in \mathcal{C}} \frac{1}{\varepsilon_\phi} \left(1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi} \right)\end{aligned}$$

The least perturbation, or at least the simplest one, consists in using the same value ε for all features. In this case, and for any positive value of ε , the first feature to require activation is the one minimizing $\frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi}$. Should there be multiple minimizers, the tie is broken by using different additional penalizations ε_ϕ . These terms can be considered as arbitrary close to zero and to each other, as long as they break the tie by yielding a unique maximizer of

$$\frac{1}{\varepsilon_\phi} \left(1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi} \right)$$

This means that an arbitrary choice can be made, on the basis of its additional penalization being slightly lower than the others'. Without expliciting them, one can state that all $(\varepsilon_\phi)_{\phi \in \mathcal{C}}$ terms are different, but sufficiently close to each other so that the factor $(1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi})$ is always dominant:

$$\arg \max_{\phi \in \mathcal{C}} \frac{1}{\varepsilon_\phi} \left(1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi} \right) = \arg \max_{\phi \in \mathcal{C}} \left(1 - \frac{\phi^\top\mathbf{\Delta}_r}{\theta_\phi} \right)$$

but in case of equality on that factor, the a priori ordering induced by ε 's breaks the tie.

This gives an interesting first rule where the perturbation terms can be dropped: in case several features are simultaneously on the edge of over-correlation, the one with largest “growth rate” ($\arg \max_{\phi \in \mathcal{C}} \frac{\phi^\top \Delta \mathbf{r}}{\theta_\phi}$) should be activated, using an arbitrary (but fixed) ordering as a second criterion if there are several of them.

Unfortunately, the same simplification can not be applied to track the zeroing of features from \mathcal{C} . When using actual perturbations, once activated, a feature's coefficient is modified with different successive rates, as others get activated, and may get zeroed and inactivated at some point. This point cannot be monitored without keeping an account of the successive modification rates and lengths, that depend on the various problem parameters and the perturbations.

Hence, the method has to be applied literally. However, as for the first method, explicitly constructing \mathcal{C} before applying the perturbations can be tedious, and there is a simpler way to get past the singularity, with a symbolic change in the parameter that results in simple rules.

3.1.2.2 Method switching

This simpler solution consists in following the proposition of (Osborne et al., 2000a) to switch from the homotopy to the active set method for a slightly lower value of λ . The interesting point is that this switch need not be explicit by actually decreasing λ and calling a different algorithm.

Let ε be the quantity by which λ is decreased. The first step of the active set method is to consider going from the starting point $\beta^* - \lambda \Delta \beta$ to the new candidate solution $\beta^* - (\lambda - \varepsilon) \Delta \beta$, and stop when a feature is zeroed. This stop is immediate, and since the direction is the one followed in the last step of the homotopy, all features that were zeroed violate the sign constraint if continuing in that direction, and thus are all inactivated.

The new unconstrained minimizer $\beta^* - (\lambda - \varepsilon) \Delta \beta$ on the shrunk active set is then computed. Note that it cannot incur new violations, if ε is sufficiently small so as not to deviate enough from the starting point to zero features that are not in \mathcal{C} . Thus, in the following, only the newly activated features may be zeroed. For this second step, and in the following ones when no more zeroing occurs, an over-correlated feature must be found and activated. Again, only features from \mathcal{C} must be considered, assuming ε is sufficiently small. Over-correlation is defined by

$$\frac{\phi^\top}{\theta_\phi} \left(\mathbf{y} - \mathbf{X}^\top (\beta^* - (\lambda - \varepsilon) \Delta \beta) \right) > \lambda - \varepsilon$$

that is

$$\frac{\phi^\top}{\theta_\phi} \left(\mathbf{y} - \mathbf{X}^\top (\beta^* - \lambda \Delta \beta) \right) - \frac{\phi^\top}{\theta_\phi} \mathbf{X}^\top \varepsilon \Delta \beta > \lambda - \varepsilon.$$

All features from \mathcal{C} being equi-correlated, this is equivalent to

$$-\frac{\phi^\top}{\theta_\phi} \mathbf{X}^\top \varepsilon \Delta \beta > -\varepsilon,$$

and

$$1 - \frac{\phi^\top}{\theta_\phi} \mathbf{X}^\top \Delta \beta > 0. \quad (3.25)$$

Not surprisingly, activating the most over-correlated feature gives the same criterion as in the previous tie-breaking method:

$$\text{activate } \arg \max_{\phi \in \mathcal{C}} \left(1 - \frac{\phi^\top \Delta \mathbf{r}}{\theta_\phi} \right)$$

However, let us not use this criterion, but rather a more practical one, since any over-correlated feature may be chosen. Multiplying inequation (3.25) by $|\theta_\phi| = \text{sign}(\theta_\phi)\theta_\phi$, we get

$$\text{sign}(\theta_\phi)(\theta_\phi - \phi^\top \mathbf{X}^\top \Delta \beta) > 0,$$

and we may use as a tie-breaking rule

$$\text{activate } \arg \max_{\phi \in \mathcal{C}} \text{sign}(\theta_\phi)(\theta_\phi - \phi^\top \Delta \mathbf{r}),$$

which is also a greatest-overcorrelation rule, but comparing the correlation c to λw rather than c/w to λ . The practical aspect is that, in the homotopy algorithm, the features that reach equi-correlation simultaneously are the members of

$$\arg \max_{\phi \in \mathcal{D}} \frac{\phi^\top \mathbf{r}^*}{\theta_\phi - \phi^\top \Delta \mathbf{r}}$$

where by definition (see (2.23)), $\theta_\phi = \text{sign}(\phi^\top \mathbf{r}^*)w(\phi)$, hence $\text{sign}(\theta) = \text{sign}(\phi^\top \mathbf{r}^*)$. Multiplying the numerator and denominator by $\text{sign}(\theta)$, the expression becomes

$$\frac{|\phi^\top \mathbf{r}^*|}{\text{sign}(\theta)(\theta_\phi - \phi^\top \Delta \mathbf{r})},$$

Since this expression, which gives the next value of λ , must be positive, it is equal to

$$\frac{|\phi^\top \mathbf{r}^*|}{|\theta_\phi - \phi^\top \Delta \mathbf{r}|},$$

Whence – assuming for the moment that no zeroing is involved – when the next-activation rule

$$\arg \max_{\phi \in \mathcal{D}} \frac{\phi^\top \mathbf{r}^*}{\text{sign}(\phi^\top \mathbf{r}^*)w(\phi) - \phi^\top \Delta \mathbf{r}}$$

gives several candidates, selecting the one with the largest absolute denominator and continuing the algorithm as is, corresponds to slightly decreasing λ and applying the Active Set method, without the need to specify this slight decrease. Note that choosing the largest absolute denominator is equivalent to choosing the lowest absolute numerator.

Concerning the zeroings, at each change of the active set, the new tentative solution is $\beta^* - (\lambda - \varepsilon)\Delta\beta$. By construction, and continuity, for all active features coming from \mathcal{C} , therefore with a coefficient zero for $\varepsilon = 0$, the coefficient is $\varepsilon\Delta\beta$. Contrarily to the previous method where these coefficients were added a term at each new active set, this one resets them at each step with this simple expression. This allows to monitor the zeroings: each time a feature is activated, the active set method takes a linear move from the previous solution to the new one, stopping at the first point where a feature is zeroed, inactivating it, and iterates until no zeroing occurs. Let $\Delta\beta$ be the regularization direction of the new solution, and $\Delta\beta'$ that of the previous one. The coefficient of an active feature ϕ from \mathcal{C} is taken from previous to its new value by

$$\beta_\phi = \varepsilon\Delta\beta_\phi' + \gamma(\varepsilon\Delta\beta_\phi - \varepsilon\Delta\beta_\phi')$$

with γ going from 0 to 1, and a zeroing occurs for

$$\gamma = -\frac{\Delta\beta_\phi'}{(\Delta\beta_\phi - \Delta\beta_\phi')}.$$

Therefore the first one occurs for

$$\begin{aligned} \arg \min_{\phi \in \sigma \cap \mathcal{C}} \left(-\frac{\Delta\beta_\phi'}{(\Delta\beta_\phi - \Delta\beta_\phi')} \right)_{\in [0,1]} &= \arg \max_{\phi \in \sigma \cap \mathcal{C}} \left(\frac{\Delta\beta_\phi'}{(\Delta\beta_\phi - \Delta\beta_\phi')} \right)_{\in [-1,0]} \\ &= \arg \min_{\phi \in \sigma \cap \mathcal{C}} \left(\frac{(\Delta\beta_\phi - \Delta\beta_\phi')}{\Delta\beta_\phi'} \right)_{\in (-\infty, -1]} \\ &= \arg \min_{\phi \in \sigma \cap \mathcal{C}} \left(\frac{\Delta\beta_\phi}{\Delta\beta_\phi'} - 1 \right)_{\in (-\infty, -1]} \\ &= \arg \min_{\phi \in \sigma \cap \mathcal{C}} \left(\frac{\Delta\beta_\phi}{\Delta\beta_\phi'} \right)_{\in (-\infty, 0]} \\ &= \arg \max_{\phi \in \sigma \cap \mathcal{C}} \left(\frac{\Delta\beta_\phi'}{\Delta\beta_\phi} \right)_{\in (-\infty, 0)} \\ &= \arg \max_{\phi \in \sigma \cap \mathcal{C}} \left(\frac{\Delta\beta_\phi'}{\Delta\beta_\phi} \right)_{\in (-\infty, 0)} \end{aligned}$$

The motivation of the interval conditions $()_{\in [0,1]}$ and its derivations is to restrict to zeroings that occur in between a solution and the next one, in the regular active set descent method. However, the features considered here are the members of

$$\arg \max_{\substack{\phi \in \sigma \\ \text{sign}(\beta_\phi^*) \neq \text{sign}(\theta_\phi)}} \left(\frac{\beta_\phi^*}{\Delta\beta_\phi} \right)_{\geq 0}$$

which, given the positivity of the ratio, verify

$$\text{sign}(\Delta\beta_\phi) \neq \text{sign}(\theta_\phi)$$

$$\text{sign}(\Delta\beta_\phi) \neq \text{sign}(\Delta\beta_{\phi'}).$$

Thus it is not needed to consider this restriction.

Let us now note that in the Active Set method, when several zeroings occur simultaneously, as is the case in the first step of the homotopy/ASD switch, they may all be zeroed at once as mentioned above, but it is also valid to zero them one by one in an arbitrary order, computing the new $\Delta\beta$ each time, and stopping when this direction does not incur any more zeroing. No cycle can occur since only zeroings are involved, and as soon as none happens anymore, it is legitimate to switch to the part of activating over-correlated features. Hence, in the first step of the switch, the criterion defined above, although it is not relevant because the coefficients are zero and not $\varepsilon\Delta\beta$, is not wrong because any order is correct.

These considerations and derivation of rules yield the fact that simply adding proper tie-breaking rules to the homotopy method is equivalent to the switch to Active Set method. The resulting method is sketched in Algorithm 10. It gives a priority to inactivation when a zeroing and an equi-correlation occur simultaneously, and priorities inside those two status changes are defined by the rules derived above. Hence, when simultaneous equi-correlations and zeroings occur, the algorithm proceeds to a sequence of inactivations and activations while λ remains unchanged (all candidates require to be activated at the current value) until all involved features agree with their status, that is the regularization direction associated to the active set does not make any inactive feature over-correlated, nor any active feature having a wrong coefficient sign. The sequence is guaranteed to converge because it corresponds to the Active Set method sequence, that repeats the following:

- while some features are zeroed following the current direction, inactivate the first one in the ASD method at $\lambda - \varepsilon$, given by the tie-breaking rule,
- if there are features requiring immediate activation, i.e. that would be over-correlated at $\lambda - \varepsilon$, activate one of them, more specifically the most over-correlated in the sense $\arg \max |correlation| - (\lambda - \varepsilon)penalization$,

The switch to and from the ASD method is seamless; it is needed only when a tie occurs and solely consists in using the tie breaking rules.

We have thus shown that, even if the number of features is greater than the number of observations, or more generally some features are linearly dependent regarding their values on the observations, the ℓ^1 penalty gives a second criterion that distinguishes between them, yielding a unique LASSO solution, unless the linear dependence is affine, which means that assigning the same coefficient to one of these features or to the equivalent combination of the others incurs the

Algorithm 10 Cycle-safe homotopy method for the regularized LASSO

Input: input vector $\mathbf{x} \in \mathcal{X}^n$, response vector $\mathbf{y} \in \mathbb{R}^n$, feature dictionary $\mathcal{D} \subset \mathbb{R}^{\mathcal{X}}$, penalization function $w : \mathcal{D} \rightarrow \mathbb{R}_+$

Output: For all $\lambda \in \mathbb{R}_+$,

$$\text{LASSO}_r(\mathbf{x}, \mathbf{y}, \mathcal{D}, w, \lambda) = \arg \min_{(\sigma, \beta) \in 2^{\mathcal{D}} \times \mathbb{R}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \beta\|_2^2 + \lambda \|\beta\|_{\mathbf{w}_\sigma}$$

Definition: For a feature function $\phi \in \mathcal{D}$, the bold notation denotes the application of ϕ to the input vector \mathbf{x} : $\phi = \phi(\mathbf{x}) = (\phi(x_1), \dots, \phi(x_n))^\top$.

- 1: $\lambda \leftarrow \infty, \sigma \leftarrow \{\}, \mathbf{X} \leftarrow [], \boldsymbol{\theta} \leftarrow ()$
- 2: **loop**
- 3: $\beta^* \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{y}$
- 4: $\Delta'_\beta \leftarrow \Delta_\beta$
- 5: $\Delta_\beta \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1}\boldsymbol{\theta}$
- 6: $\mathbf{r}^* \leftarrow \mathbf{y} - \mathbf{X}^\top \beta^*$
- 7: $\Delta_{\mathbf{r}} \leftarrow \mathbf{X}^\top \Delta_\beta$
- 8: $(\lambda_-, \phi_-) \leftarrow \max_{\substack{\phi \in \sigma \\ \text{sign}(\beta_\phi^*) \neq \text{sign}(\theta_\phi)}} \arg \max \frac{\beta_\phi^*}{\Delta_{\beta_\phi}}, \text{ break tie on } \max \frac{\Delta'_\beta \phi}{\Delta_{\beta_\phi}}$
- 9: $(\lambda_+, \phi_+) \leftarrow \max_{\phi \in \mathcal{D} \setminus \sigma} \arg \max \frac{\phi^\top \mathbf{r}^*}{\text{sign}(\phi^\top \mathbf{r}^*)w(\phi) - \phi^\top \Delta_{\mathbf{r}}}, \text{ break tie on } \min |\phi^\top \mathbf{r}^*|$
- 10: $\lambda' \leftarrow \max(\lambda_-, \lambda_+, 0)$
- 11: **output** $\forall l \in [\lambda', \lambda), \text{ LASSO}_r(\mathbf{x}, \mathbf{y}, \mathcal{D}, w, l) = (\sigma, \beta^* - l\Delta_\beta)$
- 12: **if** $\lambda' = 0$ **then terminate**
- 13: **else if** $\lambda' = \lambda_-$ **then**
- 14: $\sigma \leftarrow \sigma \setminus \{\phi_-\}$
- 15: shrink $\mathbf{X}, \beta, \boldsymbol{\theta}$ accordingly
- 16: **else if** $\lambda' = \lambda_+$ **then**
- 17: $\sigma \leftarrow \sigma \cup \{\phi_+\}$
- 18: extend \mathbf{X} by row ϕ , β by 0, and $\boldsymbol{\theta}$ by $\text{sign}(\phi^\top \mathbf{r}^*)w(\phi)$
- 19: **end if**
- 20: $\lambda \leftarrow \lambda'$
- 21: **end loop**

same cost in the coefficient's ℓ^1 norm and also the same decrease in the residual. However, this also means that the two options are really equivalent, and the choice can be made arbitrarily; this arbitrary choice is well handled by the Active Set and homotopy methods, but the CCD method uses all the affinely dependent features, splitting the needed coefficient between them.

The even more unlikely possibility of more than two simultaneous calls for activation or inactivation, of which a subset must be fulfilled, emerging from their interactions, does neither require any specific care in the Active Set method, but needs additional rules in the homotopy method to yield the same sequence of activations/inactivations as if a switch to the Active Set method was performed in a neighbouring value of λ . As unlikely as the situation may be, there is some comfort in having the insurance of handling it correctly, especially given the simplicity of the resulting adjustments. It is even more useful if this degeneracy is made less unlikely by the setting that we discuss in the following section, that is when the dictionary of features \mathcal{D} is very large, possibly not finite.

3.2 Very large feature sets

As just mentioned, one of the consequences of using very large feature sets ($\mathcal{D} \gg n$) is the increased likeliness of the degeneracies that were just discussed. That point is now well handled by our two algorithms. However, other questions arise with a large size of \mathcal{D} . No fundamental change is to be expected from using an infinite but countable set, by which we mean the problem is still well posed, its solutions exist and share the same properties whether \mathcal{D} is finite or countable. The potential issues are of practical nature: repeatedly considering an infinite number of features for activation may be problematic. Using non-countable feature sets is a greater breakthrough in the nature of the problem of which we will discuss the consequences for the different algorithms, before addressing the practical problem of using finite resources to select from infinitely many options.

3.2.1 Non-countable sets

The first considerations to make regarding the non-countability of \mathcal{D} are about the very existence of solutions, and the preservation of the sparsity property of the ℓ^1 penalty. This point was addressed in (Rosset et al., 2007), with a positive answer, under mild assumptions. This result is not surprising given the following simple arguments:

- regardless of the nature of \mathcal{D} , an infinite regularization parameter ($\lambda = \infty$) necessarily defines $\sigma = \emptyset$ as the unique LASSO minimizer;
- there exist overfitting estimators that yield a residual of $\mathbf{0}$, using at most n active features, among which some have a minimal coefficient's ℓ^1 norm, and are necessarily the minimizers of the squared residual under the corresponding constraint, because any additional active feature is a linear combination

of these n initial ones, and if activating it lessens the ℓ^1 norm, this lessening being linear, the loss decreases as long as more weight is assigned to that feature, up until another feature is zeroed, which contradicts the hypothesis that the initial solution was optimal among n -features estimators.

- the continuity and convexity properties are not affected by the nature of \mathcal{D} , which implies the existence of a regularization path between this zero solution and each of these least- ℓ^1 -norm, overfitting estimators.

3.2.1.1 Defining algorithms for arbitrary feature dictionaries

Regarding the feasibility of algorithms in this setting, a first step was done by defining the LASSO problem for arbitrary feature sets, in the first chapter, and writing the ASD and homotopy methods consequently, in the second chapter. This was a matter of notations, and simply illustrates that the nature of \mathcal{D} does not affect the essence of the homotopy and ASD methods. The CCD method, however, was presented in the common setting assuming a finite set, because it *does* rely on this assumption. However, it can be adapted to cope with arbitrary feature sets. The initial and simplest formulation of cycling through all features and updating their coefficients can be extended to different schemes. In (Friedman et al., 2010), it is mentioned that concentrating the updates on the active set yields a considerable speedup. Indeed, the cycling updates can be performed over the –finite– active set, and the remaining, inactive, features only need to be checked for overcorrelation and subsequently activated. Rather than testing/activating them through a lexicographic enumeration, it is possible to repeatedly consider the whole set \mathcal{D} and the correlation function from \mathcal{D} to \mathbb{R} , pick any over-correlated feature, and activate it with the CCD update. In this respect, the algorithm is not much different from the Active Set Descent method, as the main task is to track inactive over-correlated features and activate them, until none can be found or, in the CCD case, over-correlations become negligible. The main difference between the methods lies in the simpler updates of the CCD. Regarding the common activation part, the same arguments apply to favourize the activation of the *most* over-correlated features. This yields an alternative version of the CCD method, sketched in algorithm 11, that we call an Active Set Coordinate Descent (ASCD), to emphasize the fact that it takes into account the existence of an active set that is sparse with respect to the whole feature set, and cycles only on that active set. It is consistent with the definition of the LASSO given in Chapter 1, by not assuming a finite feature dictionary. It alternates between one cycle over the active set, and the activation of one new feature, but different schemes can be used, since both tasks can be considered as independent, and thus different frequencies in their alternation can be considered.

This algorithm, however, still bears the inconvenient, previously mentioned, of not dealing properly with affine dependencies in the feature set, and the practitioner should ensure avoiding such dependencies when designing infinite feature sets.

Algorithm 11 Active Set Coordinate Descent for the regularized LASSO

Input: input vector $\mathbf{x} \in \mathcal{X}^n$, response vector $\mathbf{y} \in \mathbb{R}^n$, finite feature dictionary $\mathcal{D} \subset \mathbb{R}^{\mathcal{X}}$, penalization function $w : \mathcal{D} \rightarrow \mathbb{R}_+$, regularization parameter $\lambda \in \mathbb{R}_+$

Output: $(\sigma, \boldsymbol{\beta})$ converges to

$$\text{LASSO}_r(\mathbf{x}, \mathbf{y}, \mathcal{D}, w, \lambda) = \arg \min_{(\sigma, \boldsymbol{\beta}) \in 2^{\mathcal{D}} \times \mathbb{R}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_{\mathbf{w}_\sigma}$$

```

1:  $(\sigma, \boldsymbol{\beta}) \leftarrow$  starting point.
2:  $\mathbf{r} \leftarrow \mathbf{y} - \mathbf{X}_\sigma^\top \boldsymbol{\beta}$ 
3: loop
4:   for  $\phi \in \sigma$  do
5:      $c \leftarrow \beta_\phi + \boldsymbol{\phi}^\top \mathbf{r}$ 
6:     if  $|c| > \lambda w(\phi)$  then
7:        $\beta_\phi \leftarrow c - \text{sign}(c) \lambda w(\phi)$ 
8:     else
9:        $\sigma \leftarrow \sigma \setminus \{\phi\}$ 
10:    end if
11:    update residual  $\mathbf{r}$ 
12:  end for
13:   $c, \phi \leftarrow \max, \arg \max_{\phi \in \mathcal{D}} \left| \frac{\boldsymbol{\phi}^\top \mathbf{r}}{w(\phi)} \right|$ 
14:   $\sigma \leftarrow \sigma \cup \{\phi\}$ 
15:   $\beta_\phi \leftarrow \boldsymbol{\phi}^\top \mathbf{r} - \text{sign}(\boldsymbol{\phi}^\top \mathbf{r}) \lambda w(\phi)$ 
16:   $\mathbf{r} \leftarrow \mathbf{r} - \beta_\phi \boldsymbol{\phi}$ 
17: end loop

```

The three algorithms now share the property of including a maximization task over \mathcal{D} , which is well defined for any set, but may not be practically feasible if \mathcal{D} is not finite. In (Rosset et al., 2007), an example is given and experimented for which this optimization can be performed for the homotopy algorithm. It consists in the set of quadratic polynomials

$$\{\phi : [0, 1] \rightarrow [0, 1], x \mapsto (x - a)^2 \mid a \in [0, 1]\}$$

for which it can be shown that the LASSO solutions can only activate features of which the knot a coincides with an observation point x_i . Thus, the optimization over \mathcal{D} can concentrate on the corresponding finite subset. However, it can be argued that such *a priori* discarding, of a similar nature as the RKHS case of *support vector machines*, simply reveals that the real expressive power of \mathcal{D} is not in adequacy with its apparent richness, and that it is a somewhat artificial, unnecessary, inflation of the finite feature set

$$\{\phi : [0, 1] \rightarrow [0, 1], x \mapsto (x - a)^2 \mid a \in \{x_1, \dots, x_n\}\}.$$

When \mathcal{D} is infinite and cannot be restricted beforehand to a finite subset, it offers a genuine rich choice that is to be determined from the data and the regularization, but this comes at the price of having to perform an optimization over a non-countable set. Before discussing the consequences and workarounds, let us give some examples of such sets and relate them to commonly used parametric regression estimators.

3.2.1.2 General-purpose infinite feature sets for low-parametric regression

Two common families of features, that are defined on $\mathcal{X} = \mathbb{R}^m$ and do not reflect strong prior assumptions on the model, are

- Gaussian radial basis functions (RBF):

$$\begin{aligned} \phi_{\Sigma, \mathbf{c}} : \mathbb{R}^m &\rightarrow [0, 1] \\ \mathbf{x} &\mapsto \frac{1}{\sqrt{(2\pi)^m |\Sigma|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{c})^\top \Sigma^{-1}(\mathbf{x} - \mathbf{c})\right) ; \end{aligned}$$

- logistic perceptrons:

$$\begin{aligned} \phi_{\mathbf{w}, b} : \mathbb{R}^m &\rightarrow [0, 1] \\ \mathbf{x} &\mapsto \frac{1}{1 + \exp(\mathbf{w}^\top \mathbf{x} + b)} . \end{aligned}$$

The RBF features are useful because they are based on a well-formed measure of closeness between a centre \mathbf{c} and the input \mathbf{x} . It is well-formed because, as mentioned in chapter 1, the Gaussian function is ubiquitous in nature and has nice mathematical properties, mostly for the same reasons. They are parameterized by the values noted in subscript in their definition: a centre \mathbf{c} and a covariance

matrix Σ . A classical way to use them in regression consists in the experimenter selecting a set of these features beforehand or through repeated trials, by choosing the parameters, and then performing a –possibly regularized– least-squares on this finite set. The most classical and less informative way to select the features is to use a grid of centres in the input sets, and a fixed diagonal covariance matrix such that the bandwidth of the features match the mesh spaces of the grid. Another is to use the input observed points as centres, and also a fixed covariance matrix. The use of this common matrix Σ gives $k(\mathbf{x}, \mathbf{x}') = \phi_{\Sigma, \mathbf{x}'}(\mathbf{x})$ the reproducing kernel property and justifies the choice of the observed inputs as centres. In the general case, several different covariance matrix / bandwidths may naturally be used together. These approaches can be qualified as *parametric*. The term of *parametric regression* and its counterpart *non-parametric regression* are sometimes given different meanings depending on the scientific community, authors, or context, and it is worth trying to clarify this taxonomy.

One usual understanding of these terms is that parametric regression consists in defining the estimator independently of the observations –but depending on the nature of the data– by an expression in which a fixed number of numerical variables (the parameters) are to be defined from the observations, and methods in which more than the predefined numerical parameters is determined by the observations qualifies as non-parametric. In other words, parametric regression designates the *most* parametric settings in which there is a strict separation between defining the structure before the observations and fitting the fixed number of parameters from these observations. In this view, kernel methods like Gaussian Processes are considered non-parametric, because the number of features as well as the features themselves ($k(x_1, \cdot), \dots, k(x_n, \cdot)$) are determined by the the observations. It should be noted, however, that the influence of the data on the structure is limited, by the fact that it is automatic, straightforward, and most importantly taking into account only the predictor variable. Such methods actually incorporate a distinct intermediate step between choosing the structure given the nature of the data, and fitting the parameters given the observations. The last step takes into account the empirical joint distribution between X and Y , the relation between these two being the core information to extract, whereas the additional intermediate step simply, but fruitfully, adapts the structure to the marginal distribution of X . The structure is still strongly defined beforehand by the choice of the kernel, and the observations are still processed to fit numerical parameters, but kernel methods can be qualified as non-parametric, in this first understanding of the term.

Another definition of non-parametric regression is in some sense the opposite of the first one. It is probably more legitimate, having a clear and simple definition and being included in the more general concept of non-parametric statistics. Rather than defining as parametric the most parametric methods and non-parametric the other ones, it simply –and logically– defines as non-parametric regression the methods in which no parameter is defined, and the observations are used “as is” for the estimator to make a prediction, rather than being processed,

compiled, into such parameters. Only a few practical estimators are genuinely non-parametric, according to this definition, and consist in interpolating between the observations or using weighted averages of the observed responses. However, kernel methods present a strong similarity to those: in order to define the estimator, they do –and must– store all observations, or more precisely the associated features $k(x_i, \cdot)$, and this can be thought of as the main point of the definition. Thus, even if the coefficients to these features are computed in the very same way as the parameters of parametric regression, kernels methods may also be thought as non-parametric in this respect.

Given the coexistence of different definitions, both subject to interpretation, we advocate the use of the term *low-parametric* regression for methods that do compute parameters such as linear coefficients, but in which the structure (including the number of these parameters, and nonlinear parameters) is also, more or less, defined from the observations. Let us also, in order to classify more precisely among these methods, identify the main components that define a parametric estimator: assuming $\mathcal{Y} = \mathbb{R}$, a parametric regression method ultimately defines a function $\mathcal{X} \rightarrow \mathbb{R}$ of which the only variable is the input for which a response is to be predicted. It does so by defining a general operator, in which other variables are to be defined either by the experimenter (hyper-parameters), or from the observations, either from their marginal distributions (super-parameters), or the joint one (parameters). These three types of parameters can be either structural (e.g. number of features), or numerical/symbolical. A first point of comparison between methods is how much the general operator is restrictive by itself, or how much expressiveness is given, to be tightened by the choice of parameters. A second point is how the parameters are distributed among hyper, super, and simple ones, giving an idea of how low-parametric they are.

In (Loth et al., 2009), we proposed the use of the homotopy method over infinite feature sets. The method, named ECON for equi-correlation networks, uses as a feature dictionary a parametric space of feature functions, two natural examples being the set of all logistic perceptrons (named ECONN for *Neural Networks*)

$$\mathcal{D} = \left\{ \mathbf{x} \mapsto \frac{1}{1 + \exp(\mathbf{w}^\top \mathbf{x} + b)} \mid \mathbf{w} \in \mathbb{R}^m, b \in \mathbb{R} \right\},$$

and the set of all Gaussian features (ECORBF)

$$\mathcal{D} = \left\{ \mathbf{x} \mapsto \frac{1}{\sqrt{|\Sigma|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{c})^\top \Sigma^{-1}(\mathbf{x} - \mathbf{c})\right) \mid \mathbf{c} \in \mathbb{R}^m, \Sigma \in \mathbb{R}^{m \times m} \right\}.$$

The consequences of such a choice for \mathcal{D} and the practical aspects are discussed in sections 3.2.1.4 and 3.2.2. We may however mention here a few first simplifications that ease the task of selection among these. Firstly, the possible values for the features parameters may be restricted to specific intervals: for the Gaussian features, the range of meaningful values for the centres and bandwidths can be derived from the range of the observations, and, provided that the inputs are normalized, the weights of perceptron can safely be constrained in some interval.

Secondly, allowing a total freedom for the covariance matrix Σ may incur more difficulties than benefits. A dimensionality of $O(m \times m)$ for \mathcal{D} may complicate the search over it while not bringing much improvement other than a capacity to over-fit the data. Hence one may restrict \mathcal{D} to the family of Gaussian features with diagonal covariance matrix, that is

$$\Sigma = \begin{pmatrix} \sigma_1^2 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \sigma_m^2 \end{pmatrix}$$

which gives

$$\mathcal{D} = \left\{ \mathbf{x} \mapsto \frac{1}{\|\boldsymbol{\sigma}\|_2} \exp \left(-\frac{1}{2} \sum_{i=1}^m \left(\frac{x_i - c_i}{\sigma_i} \right)^2 \right) \mid \mathbf{c} \in \mathbb{R}^m, \boldsymbol{\sigma} \in \mathbb{R}^m \right\}.$$

One may even reduce the number of covariance parameters to one, with $\Sigma = \sigma^2 \mathbf{I}$, that is $\sigma_1 = \dots = \sigma_m = \sigma$, which we may call an isovariant RBF and gives

$$\mathcal{D} = \left\{ \mathbf{x} \mapsto \frac{1}{\sigma\sqrt{m}} \exp \left(-\frac{\|\mathbf{x} - \mathbf{c}\|_2^2}{2\sigma^2} \right) \mid \mathbf{c} \in \mathbb{R}^m, \sigma \in \mathbb{R} \right\}.$$

The last simplification is common in RBF regression, since most of the methods, by necessity or for practicality, allow only one or a few possible values for the shape of the features, as hyper-parameters; in the absence of a prior knowledge, an uninformative symmetric shape is a natural choice, after normalizing the observations in each dimension, and leaves a single hyper-parameter to be set for the range of the features.

This is the case in classical RBF regression, where a fixed number of isovariant RBF are defined, of which the centres span the input space, generally in a grid form. A simple least-squares regression is then performed on these features. Kernel methods like Gaussian processes ((Rasmussen and Williams, 2006)) or Support Vectors algorithms ((Smola and Schölkopf, 2004)) rely on the transformation of the input space into the feature space by one kernel. This results in the use of n identical features in terms of shape, centred (in the case of a Gaussian kernel) on each observation. The use of multiple kernels, of which the combination adapts to the learning data, has been proposed in (Lanckriet et al., 2004), (Bach et al., 2004), (Qiu and Lane, 2005), or (Bach, 2008). In these methods, the different kernels are structured together, either by being combined into a sum of which the coefficients are learnt, or consisting in the decomposition of one original kernel. These works place themselves in the kernel framework, and ultimately select a unique kernel function that forms the regression features by being applied to each observed input.

In contrast with these grid and kernel approaches, the simple application of the LASSO to a parameterized space of features leaves the most parameters to be really determined from the observations, in an integrated way (the choice

Table 3.1: Comparison of different regression methods in terms of parameters.

The three approaches share the same initial framework of modelling the data by a linear combination of Gaussian features, and differ in the way the estimator is determined from the data. Hyper-parameters must be set independently of the method itself, by knowledge or validation methods. Super-parameters are determined by the form of the observations data, without taking into account the X/Y relationship. Parameters fit the data.

	<i>Structural</i>	<i>Numerical</i>
Fixed radial basis functions networks		
<i>hyper-parameters</i>	number of features	centres, covariance matrices
<i>super-parameters</i>	–	–
<i>parameters</i>	–	linear coefficients
Gaussian kernel methods		
<i>hyper-parameters</i>	–	covariance matrix
<i>super-parameters</i>	number of features	centres
<i>parameters</i>	–	linear coefficients
ECORBF		
<i>hyper-parameters</i>	–	regularization trade-off λ , covariance matrix form
<i>super-parameters</i>	–	–
<i>parameters</i>	number of features	centres, covariance matrices elements, linear coefficients

or optimization of the features or kernel is not separated from the computation of the linear coefficients). Table 3.1 summarizes the different natures of the parameters in methods that all compute an estimator in the form of a linear combination of Gaussian features. The idea behind making no initial restrictions on the features other than, e.g. the Gaussian family, and letting all parameters fit the observations, is that the ℓ^1 -norm penalization could be sufficient to both select a subset of features and prevent overfitting the observations, which are correlated tasks. In other words, there should be no need to restrict the space of estimators beforehand by hyper-parameters, since the ℓ^1 selection is more fitted to the data, while the sole sparsity of this selection should prevent overfitting by itself.

An other point of comparison can be made with the feed-forward artificial neural networks (ANN). In their simple form of one hidden layer, they provide estimators that are linear combinations of perceptron features as defined at the beginning of this section. The method takes a somewhat similar approach as ECON, by optimizing together, and from the data, the linear and nonlinear parameters. The classical ANN approach fixes the number k of features (hidden units) and proceeds to a global optimization over these parameters, by following stochastic gradient directions over them. This optimization is thus performed on

a space of dimension $k(m+1) + k$. The ECONN approach replaces the hyperparameter k by the equivalent regularization parameter λ , and an important difference is that, by following the regularization path, a sequence of estimators for different values of λ is produced, among which an informed choice can be made. However, the main difference between the two approaches lies in the dimensionality of the optimizations: a global optimization over the $O(km)$ parameters is replaced by a sequence of optimizations over the $O(m+1)$ parameters of *one* feature.

3.2.1.3 Biasing the regularization path

As was just mentioned, the sole ℓ^1 or ℓ^0 penalization is able to prevent or limit overfitting, because the most effective way to reduce the residual with such a constraint is to select features that fit the global shape of the observations. A gradual relaxation of the constraint by following the regularization path thus produces a sequence of estimators, that models more and more precisely the observations. Provided that the model and the noise (or expectation and variance) have distinct scales, a reasonable model is likely to be found at some point in this sequence. However, sparsity in itself is not a direct, well-formed measure or criterion for model recovery.

The normalization of the features, explained and motivated in section 2.6.1, ensures a somewhat fair selection among features, based on shape-fitting. However, this also means that a feature with a small support on \mathcal{X} (e.g. a Gaussian with small bandwidth σ), might be selected rapidly on the regularization path, if it happens to fit the observations especially well on this support. Whence, it should be profitable to introduce an additional criterion of regularity and generalization property. Such a criterion is the preference for large-support features: a feature that has significant values on a large number of observations cannot fit a noise that consists in independent variations from point to point. Thus if the features are penalized with respect to such a criterion, it should reinforce the property of the regularization path to go from model fitting to noise fitting. In figure 2.5, one can see that on the sinusoidal problem of section 2.6.1, the first feature selected by the *normalized LASSO* (that is, features are simply normalized), though fitting the general shape of the model, does not cover the widest sinusoidal wave, but the narrowest, because the shapes of the Gaussian and the observations are closer on this region. The next few steps successively match the other waves, then a second phase, adjusts more precisely to the overall shape, before small support variations, corresponding to the noise, are fitted. Intuitively and empirically, the last two phases tend to overlap, as small adjustments of a smooth estimator and local noise fitting must have the same value in terms of residual reduction and coefficient cost.

In order to bias the regularization path towards an overall decrease in the support of selected features, it suffices, in the case of Gaussian features, to use penalization factors that decreases with the size of the support. This size is

properly expressed by the square root of the determinant of the covariance matrix Σ , that is $\prod_{i=1}^m \sigma_i$ in the diagonal case, and σ in the isovariant case; let us simply denote $\sigma = \sqrt{\det(\Sigma)}$ in the general case. The penalization function can thus be set to

$$w(\phi_{\mathbf{c}, \Sigma}) \propto 1/\sigma$$

after normalization, or to

$$w(\phi_{\mathbf{c}, \Sigma}) \propto \|\phi_{\mathbf{c}, \Sigma}\|_2 / \sigma$$

for handling both the normalization and the differentiated penalization.

If one wants to emphasize penalization of features with very small supports, that is of the order of the density of the observation points, and in the absence of a knowledge or regularity of this density, a reasonable choice seems to be a logarithmic penalization function:

$$w(\phi_{\mathbf{c}, \Sigma}) \propto \frac{1}{\log(1 + \sigma)},$$

which makes the “favourization” (inverse of penalization) decrease regularly as σ decreases for most values, and decrease more and more rapidly as σ tends to zero.

Figures 3.1 and 3.2 display the same plots as figure 2.5, when using those penalizations functions. The first feature to be selected in both cases has a very large bandwidth, and almost corresponds to a linear least-squares (on \mathcal{X}). For the $1/\sigma$ penalization, the first estimator to cover all waves of the sinusoidal function is also the best one in terms of true loss, and in the $1/\log(1 + \sigma)$ case, it is almost optimal. This is a major difference with the case of simple normalization, for which the very first steps immediately cover the global shape, and are followed by a long phase of adjustments, that are subject to corruption by noise-fitting elements. The precise behaviour of the three penalization options can be seen in figure 3.3, that shows the evolution of the squared residual (empirical loss) and a “true” loss (with respect to the unnoisy values), with the number of steps of the homotopy (i.e. the number of break points on the regularization path). The differentiated-penalizations paths yield better estimators, and exhibit clearly distinct phases for model-fitting and noise-fitting: each drop in the empirical loss corresponds to the first fit of one wave, and is followed by adjustments, before the next region is treated. Hence, once the last region is covered, the estimator is already optimal or quasi-optimal, and the remaining steps fit the noise. This last phase is easily detectable from the empirical loss, as starting from or near after the last drop. Also, the range in which satisfyingly good estimators are found is large (steps 135 to 225 for $1/\sigma$ penalization), whereas the optimal estimator for normalized features appears and deteriorates very quickly, and does not leave clear evidences on the empirical loss. A final observation on these paths is that the benefits of differentiated penalizations comes at the price of a larger number of steps, and the $1/\log(1 + \sigma)$ option, although it does not yield the expected

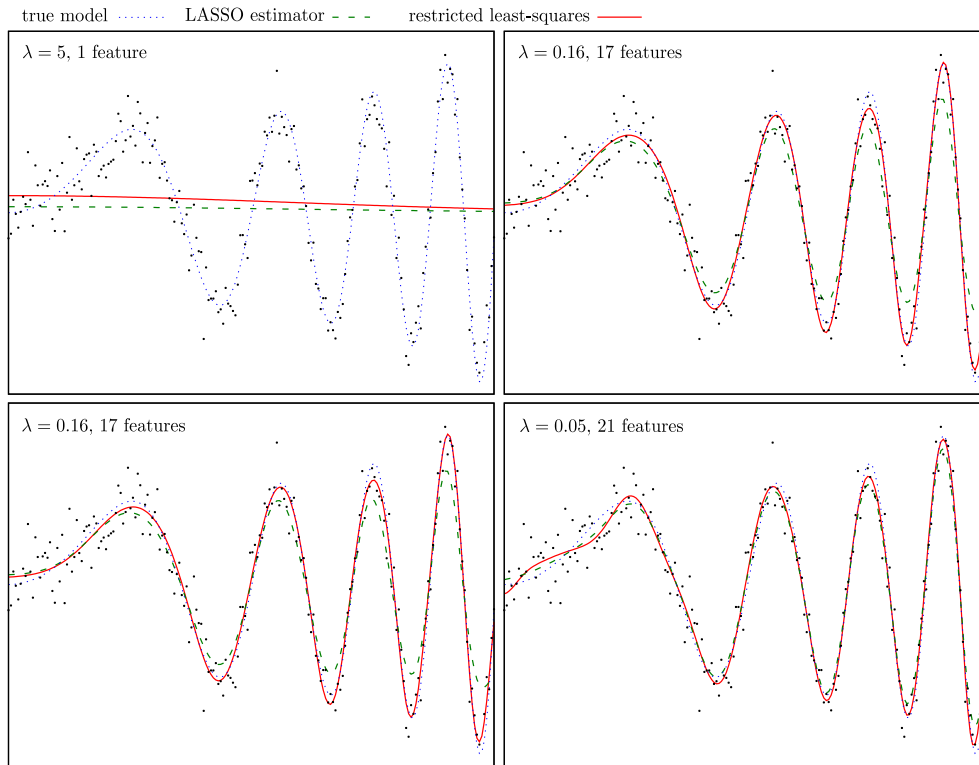


Figure 3.1: LASSO and restricted-least-squares estimators for different values of λ , for noisy observations of a sinusoidal function, and 10000 Gaussian features, with features penalizations inversely proportional to their bandwidth σ .

The four estimators are, successively, the first one with one feature, the first to approximately fit all sinusoidal waves, the closest to the true model (the latter two being the same here), and the first to be noticeably overfitting.

improvement in terms of delaying overfitting, gives quite similar results as $1/\sigma$ with twice less steps.

3.2.1.4 Optimizations over infinite sets

A strict optimization of a continuous is naturally not feasible in finite time by deterministic computing machines, and one has to rely on approximate optimizations. The consequences on the three algorithms are of different natures, the homotopy method being more affected than the others. Indeed, the maximization is a mandatory part of the algorithm, at each step: the exact regularization path is followed, and assumed to have been correctly followed when determining the next break point, meaning the exact maximizer over ϕ of

$$\frac{\phi^T \mathbf{r}^*}{\text{sign}(\phi^T \mathbf{r}^*)w(\phi) - \phi^T \Delta \mathbf{r}} \quad (3.26)$$

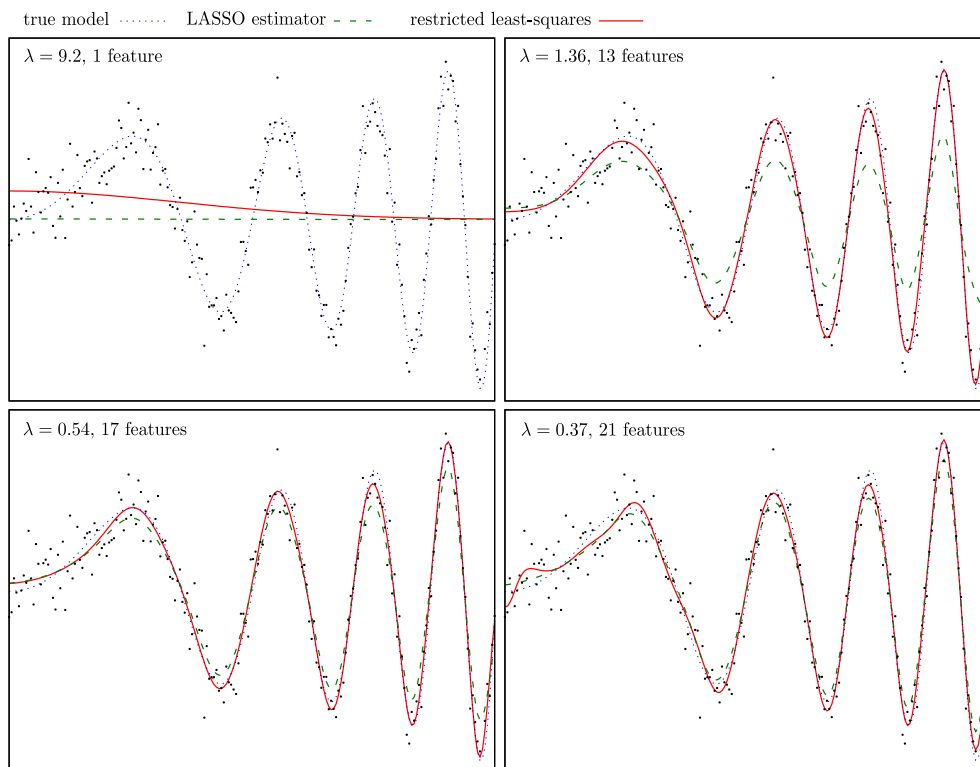


Figure 3.2: LASSO and restricted-least-squares estimators for different values of λ , for noisy observations of a sinusoidal function, and 10000 Gaussian features, with features penalizations inversely proportional to $\log(1 + \sigma)$.

The four estimators are, successively, the first one with one feature, the first to approximately fit all sinusoidal waves, the closest to the true model, and the first to be noticeably overfitting.

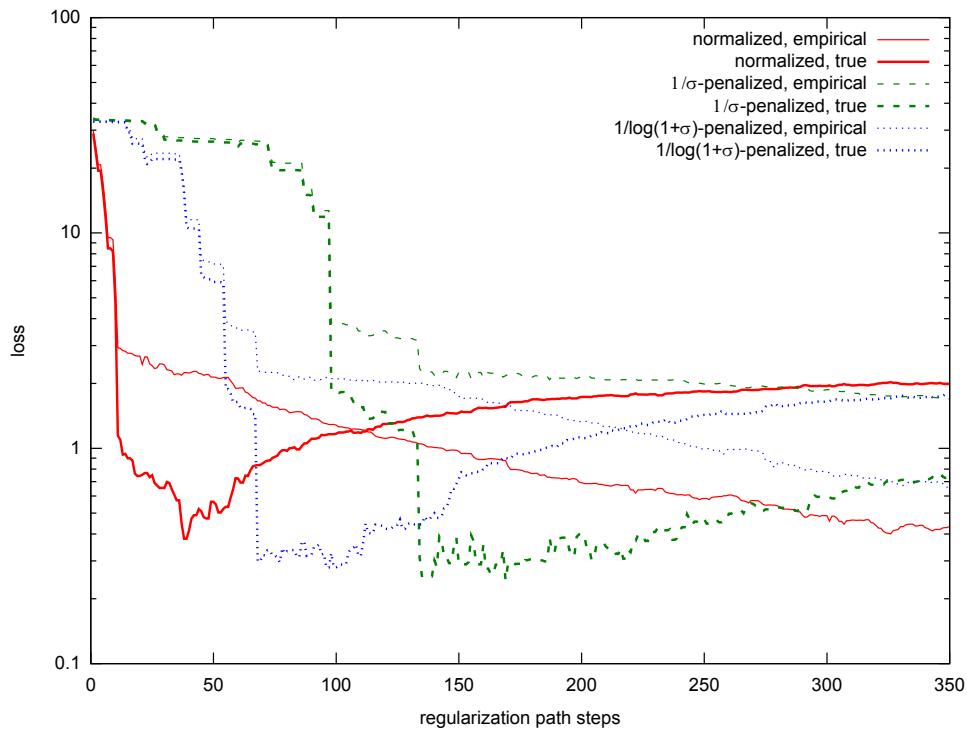


Figure 3.3: For the sinusoidal problem of figures 2.5, 3.1 and 3.2, corresponding to different feature penalizations, plot of the squared residual between the restricted-least-squares estimator and, respectively, the noisy outputs (empirical loss) and the model (true loss), against each step of the regularization path.

The bandwidth favourizations do achieve better true losses, at the price of a larger number of steps. Most importantly, an optimal or quasi-optimal estimator is obtained just at the end of the first phase that fits each wave of the model, which can be detected from the evolution of the empirical loss.

must be found. If it fails to be activated, it becomes, by definition, over-correlated. The expression (3.26), that gives the value of λ at which equi-correlation will occur, is then not relevant to detect, in subsequent steps, that a feature was missed and is now over-correlated. Hence a first adaptation of the homotopy algorithm, when an infinite dictionary and approximate optimization is used, consists in explicitly detecting over-correlated features ($|\phi^\top \mathbf{r}| > \lambda w(\phi)$) in the optimization process. Secondly, one must choose what to do with such features, in order to get back on the regularization path. One possibility, adopted in (Loth et al., 2009) consists in adapting the problem to the solution, by making the feature *not* over-correlated, by means of tuning its penalization:

$$\mathbf{if} \ |\phi^\top \mathbf{r}| > \lambda w(\phi) \ \mathbf{then} \ w(\phi) \leftarrow \frac{\phi^\top \mathbf{r}}{\lambda}$$

This makes the feature equi-correlated, thus making the current solution valid for the distorted problem. If it were not activated, it could evolve in both ways, becoming over-correlated or under-correlated, and in principle, in the second case, it need not be activated. However, it is simpler to activate it, and let the second case translate into its coefficient taking the wrong sign, and its inactivation. This activation is immediate, meaning λ is unchanged: contrarily to the normal case where one computes for which value of λ a feature is entering the solution, this is a correction to the LASSO solution for the current value of λ . Thus the algorithm either activates an over-correlated feature or, if none is found, moves on the regularization path up to the point where a feature enters or leaves the LASSO solution.

The alteration of penalization for over-correlated features should not affect the result to too great an extent, if the optimization is not too loosely approximated and detections of over-correlations do not occur too late. This can be understood by noting that this procedure is a soft version of a more radical one that consists in simply removing such “missed” features from the dictionary. A reasonably approximated optimization is valid, i.e. exact, for a large subspace of \mathcal{D} , and removing a few points or parts that were wrongfully ignored, leaves an alternative feature set that is still rich and on which the optimization was exact. The re-penalization procedure is even less perturbing, because in the presence of a missed feature, rather than stating it was not part of \mathcal{D} , it is simply considered to be more penalized than others; and the sooner it is detected, the lower it is over-correlated, and the lower the alteration. A non-countable feature dictionary is typically continuous in its parameters, as in the examples of Gaussian and perceptron families, and so is the correlation of its features. Hence missing a feature in the homotopy algorithm yields overcorrelation for a continuous, radial, region of \mathcal{D} , of which the radius tends to grow, if the overcorrelation increases. This makes this region more and more easy to detect. When it is detected, identifying the local maximizer of correlation, and activating it after rescaling its penalization, will generally make the correlation of its neighbourhood decrease, causing no further complications to the algorithm.

The sketch of this adaptation of the homotopy method to approximate optimization is given in Algorithm 12.

Another possibility when encountering an over-correlated feature is naturally to switch to the active set descent method, which by essence activates over-correlated features until there are none anymore. The procedure for such a feature would then be to activate it, compute the non-sign-constrained solution, and shrink the active set while this solution is not sign-compliant. The resulting algorithm is described in Algorithm 13. However, considering this algorithm, one might question its utility with the following argument: we have incorporated an active set descent procedure to catch up from the drifts of an imperfect, approximate, homotopy algorithm, but is it really useful, if the active set descent is used, to mix it with the homotopy? What are the benefits of an approximate homotopy compared to a simple active set descent? The purpose of the homotopy is to follow the exact regularization path, including all changes of the active set as λ decreases, as opposed to the active set descent for which a predefined sequence of λ is chosen, and the solutions are computed using one as a warm start from the next one. We have seen that, empirically, the number of changes of the active set are equivalent in both methods. In the regular case of a finite dictionary, the homotopy offers the advantage that each of these changes of active set correspond to the actual solutions on the regularization path. However, in the infinite dictionary case, this advantage, in addition to not presenting a strong practical interest, is not strictly provided (one drifts from the regularization path if optimization is not exact), and comes at a higher price: in addition to optimizing the overcorrelation function (when would each feature become over-correlated as λ decreases), one has to track the current-correlation function to check for already over-correlated features. In contrast with this, a simple active set descent only considers the correlation function, and there is no ill consequences of not picking the *most* over-correlated feature: the order in which these features are activated is not mandatory, it is only when there appears to be no more of them that an error (there *are* some more over-correlated features) yields an approximate solution rather than the exact one. This solution is nevertheless a reasonably good one, and a good warm start for the next one.

3.2.1.5 Nonlinear regularization paths

In addition to the use of an approximate optimization, an other possible consequence of using a continuous feature set is to change the nature of the regularization path. It was shown to be piecewise linear (in the coefficient's space), with a finite number of segments, as a consequence of the finiteness of \mathcal{D} . Thus when \mathcal{D} is not finite, it might occur that the number of segments is infinite, and the path can have nonlinear portions, meaning that for some interval of values of λ , any different value yields a different active set. Thus an hypothetical exact homotopy algorithm would require an infinite number of steps. This is not an issue in practical cases, e.g. when using Gaussian or perceptron families, because

Algorithm 12 ECON: approximate homotopy method for the regularized LASSO, with adaptive penalization

Input: input vector $\mathbf{x} \in \mathcal{X}^n$, response vector $\mathbf{y} \in \mathbb{R}^n$, feature dictionary $\mathcal{D} \subset \mathbb{R}^{\mathcal{X}}$, penalization function $w : \mathcal{D} \rightarrow \mathbb{R}_+$,

Output: For all $\lambda \in \mathbb{R}_+$,

$$\widehat{\text{LASSO}}_r(\mathbf{x}, \mathbf{y}, \mathcal{D}, w, \lambda) \approx \arg \min_{(\sigma, \beta) \in 2^{\mathcal{D}} \times \mathbb{R}} \|\mathbf{y} - \mathbf{X}_\sigma^\top \beta\|_2^2 + \lambda \|\beta\|_{\mathbf{w}_\sigma}$$

1: $\lambda \leftarrow \infty, \sigma \leftarrow \{\}, \mathbf{X} \leftarrow [], \boldsymbol{\theta} \leftarrow ()$

2: **loop**

3: $\beta^* \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1} \mathbf{X}\mathbf{y}$

4: $\Delta'_\beta \leftarrow \Delta_\beta$

5: $\Delta_\beta \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1} \boldsymbol{\theta}$

6: $\mathbf{r}^* \leftarrow \mathbf{y} - \mathbf{X}^\top \beta^*$

7: $\Delta_{\mathbf{r}} \leftarrow \mathbf{X}^\top \Delta_\beta$

8: $\mathbf{r} \leftarrow \mathbf{r}^* + \lambda \Delta_{\mathbf{r}}$

9: $(\lambda_-, \phi_-) \leftarrow \max_{\phi \in \sigma} \arg \max_{\text{sign}(\beta_\phi^*) \neq \text{sign}(\theta_\phi)} \frac{\beta_\phi^*}{\Delta_{\beta_\phi}}, \text{ break tie on } \max \frac{\Delta'_{\beta_\phi}}{\Delta_{\beta_\phi}}$

10: $(\lambda_+, \phi_+) \approx \max_{\phi \in \mathcal{D}} \arg \max \frac{|\phi^\top \mathbf{r}|}{w(\phi)} \quad \triangleright \text{ approximate maximization}$

11: **if** $\lambda_+ < \lambda$ **then** $(\lambda_+, \phi_+) \approx \max_{\phi \in \mathcal{D}} \arg \max \frac{\phi^\top \mathbf{r}^*}{\text{sign}(\phi^\top \mathbf{r}^*) w(\phi) - \phi^\top \Delta_{\mathbf{r}}}$

12: **break tie on** $\min |\phi^\top \mathbf{r}^*|$

13: $\lambda' \leftarrow \max(\lambda_-, \lambda_+, 0)$

14: **output** $\forall l \in [\lambda', \lambda), \text{ LASSO}_r(\mathbf{x}, \mathbf{y}, \mathcal{D}, w, l) \approx (\sigma, \beta^* - l \Delta_\beta)$

15: **if** $\lambda' = 0$ **then terminate**

16: **else if** $\lambda' = \lambda_-$ **then**

17: $\sigma \leftarrow \sigma \setminus \{\phi_-\}$

18: **shrink** $\mathbf{X}, \beta, \boldsymbol{\theta}$ accordingly

19: **else if** $\lambda' = \lambda_+$ **then**

20: $\sigma \leftarrow \sigma \cup \{\phi_+\}$

21: **if** $\lambda_+ > \lambda$ **then** $\theta \leftarrow \frac{\phi_+^\top \mathbf{r}}{\lambda}$ **else** $\theta \leftarrow \text{sign}(\phi_+^\top \mathbf{r}^*) w(\phi_+)$

22: **extend** \mathbf{X} by row ϕ , β by 0, and $\boldsymbol{\theta}$ by θ

23: **end if**

24: **if** $\lambda' < \lambda$ **then** $\lambda \leftarrow \lambda'$

25: **end loop**

Algorithm 13 Mixed algorithm: approximate homotopy method for the regularized LASSO, with corrections by active set descent.

```

1:  $\lambda \leftarrow \infty, \sigma \leftarrow \{\}, \mathbf{X} \leftarrow [], \boldsymbol{\theta} \leftarrow ()$ 
2: loop
3:    $\boldsymbol{\beta}^* \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{y}$ 
4:    $\Delta'_\beta \leftarrow \Delta_\beta$ 
5:    $\Delta_\beta \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1}\boldsymbol{\theta}$ 
6:    $\mathbf{r}^* \leftarrow \mathbf{y} - \mathbf{X}^\top\boldsymbol{\beta}^*$ 
7:    $\Delta_\mathbf{r} \leftarrow \mathbf{X}^\top\Delta_\beta$ 
8:    $\mathbf{r} \leftarrow \mathbf{r}^* + \lambda\Delta_\mathbf{r}$ 
9:    $(c_{\max}, \phi_c) \approx \max_{\phi \in \mathcal{D}} \arg \max \frac{|\phi^\top \mathbf{r}|}{w(\phi)} \quad \triangleright \text{approximate maximization}$ 
10:  if  $c_{\max} > \lambda$  then
11:     $\sigma \leftarrow \sigma \cup \{\phi_c\}$ 
12:    extend  $\mathbf{X}$  by row  $\phi_c$ ,  $\boldsymbol{\beta}$  by 0, and  $\boldsymbol{\theta}$  by  $\text{sign}(\phi_c^\top \mathbf{r})w(\phi_c)$ 
13:    repeat
14:       $\bar{\boldsymbol{\beta}} \leftarrow (\mathbf{X}\mathbf{X}^\top)^{-1}(\mathbf{X}\mathbf{y} - \lambda\boldsymbol{\theta})$ 
15:       $(\gamma, \phi) \leftarrow \min, \arg \min_{\phi \in \sigma} \left( \frac{\beta_\phi}{\beta_\phi - \bar{\beta}_\phi} \right)_+$ 
16:      if  $\gamma < 1$  then
17:         $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \gamma(\bar{\boldsymbol{\beta}} - \boldsymbol{\beta})$ 
18:         $\sigma \leftarrow \sigma \setminus \{\phi\}$ 
19:        shrink  $\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\theta}$  accordingly
20:      end if
21:    until  $\gamma \geq 1$ 
22:  else
23:     $(\lambda_-, \phi_-) \leftarrow \max_{\substack{\phi \in \sigma \\ \text{sign}(\beta_\phi^*) \neq \text{sign}(\theta_\phi)}} \arg \max \frac{\beta_\phi^*}{\Delta_{\beta\phi}}, \text{ break tie on } \max \frac{\Delta'_{\beta\phi}}{\Delta_{\beta\phi}}$ 
24:     $(\lambda_+, \phi_+) \approx \max_{\phi \in \mathcal{D}} \arg \max \frac{\phi^\top \mathbf{r}^*}{\text{sign}(\phi^\top \mathbf{r}^*)w(\phi) - \phi^\top \Delta_\mathbf{r}}$ 
25:    break tie on  $\min |\phi^\top \mathbf{r}^*|$ 
26:     $\lambda' \leftarrow \max(\lambda_-, \lambda_+, 0)$ 
27:     $\forall l \in [\lambda', \lambda), \text{ LASSO}(\mathbf{x}, \mathbf{y}, \mathcal{D}, l) \leftarrow (\sigma, \boldsymbol{\beta}^* - l\Delta_\beta)$ 
28:    if  $\lambda' = 0$  then terminate
29:    else if  $\lambda' = \lambda_-$  then
30:       $\sigma \leftarrow \sigma \setminus \{\phi_-\}$ 
31:      shrink  $\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\theta}$  accordingly
32:    else if  $\lambda' = \lambda_+$  then
33:       $\sigma \leftarrow \sigma \cup \{\phi_+\}$ 
34:      extend  $\mathbf{X}$  by row  $\phi$ ,  $\boldsymbol{\beta}$  by 0, and  $\boldsymbol{\theta}$  by  $\text{sign}(\phi^\top \mathbf{r}^*)w(\phi)$ 
35:    end if
36:     $\lambda \leftarrow \lambda'$ 
37:  end if
38: end loop

```

neighbouring features are similar ($\phi_{\Sigma,c}(\mathbf{x})$ or $\phi_{\mathbf{w},b}(\mathbf{x})$ are continuous w.r.t. their parameters), and the optimality of one against its neighbours is more than likely to remain within a certain range of values for λ . This can be thought as the space of features behaving by zones, that evolve with λ . Once a local optimum of the function to optimize is activated, this function (the correlation, in the case of ASD) is decreased for the whole zone, which is then globally discarded for the next steps. However, it is possible to define an ill-posed problem that yields a degenerate situation in which any infinitesimal change in λ yields the inactivation of a feature in profit of one of its immediate neighbour. This is the case in the following problem:

- $n = 2$,
- $y_1 = y_2 = 1$,
- \mathcal{D} is a continuum of features parameterized by $\theta \in [0, 1]$, for which, when θ decreases, the direction of a feature tends to that of the target \mathbf{y} , while its ℓ^2 -norm decreases:

$$\mathcal{D} = \{\phi_\theta, \phi_\theta(x_1) = \theta, \phi_\theta(x_2) = \theta(1 - \theta) \mid \theta \in [0, 1]\}$$

The problem's data is illustrated in figure 3.4, in which it can be seen that ϕ_1 , although forming the biggest angle with \mathbf{y} , is the most correlated, because of its greater norm. When activated and given any positive coefficient β , the target is changed to $\mathbf{y} - \beta\phi_1$, to which the largest correlation is then achieved by another feature. The activation of the latter disqualifies ϕ_1 , for the optimal coefficient direction for the two gives a negative sign to ϕ_1 , and it is thus immediately inactivated. The same phenomenon is then repeated, until the two coefficients are and remain positive until the least-squares solution. Thus, when discretizing \mathcal{D} by using a finite sequence for the values of θ , the regularization path successively includes each feature and discards the previous one, and in the limit where \mathcal{D} is continuous, the active set changes continuously with λ . Figure 3.5 shows the plots of the coefficient's ℓ^1 -norm against λ , on which the linearity of path segments also appears, for different discretizations of \mathcal{D} and for the whole set, for which the path becomes nonlinear.

3.2.2 Practical optimization

Any optimization procedure over an infinite set can naturally consider only a finite subset of points in practise. In the present case, and especially for parameterized RBF, a first discretization step that restricts the possible values to a regular grid on the parameter space can be done harmlessly, if the grid is sufficiently dense. Indeed, the features being continuous in their parameters, one feature has equivalent relevance as the neighbouring ones, in terms of correlations, which are also continuous in the parameters. This may seem to discard the above considerations about infinite feature sets as unuseful. However, a grid

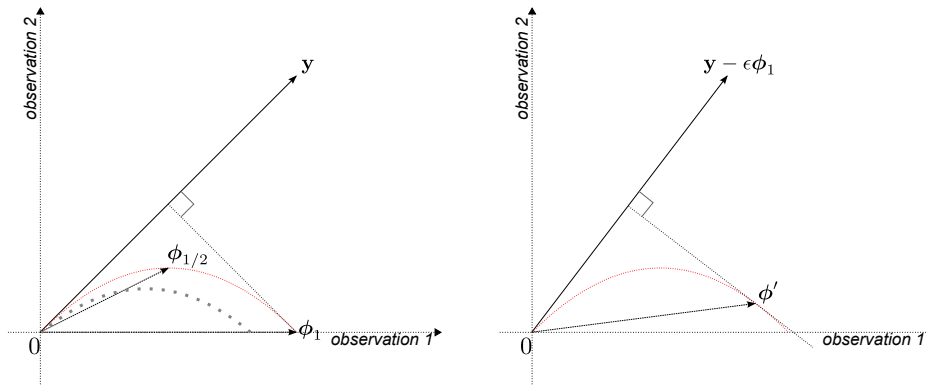


Figure 3.4: A problem, described in section 3.2.1.5, for which the active set of the LASSO solution evolves continuously with λ .

ϕ_1 , though forming the largest angle with \mathbf{y} , has the highest correlation with it, as seen from its orthogonal projection. It is thus the first to be activated, for $\lambda = 1$. As soon as a positive coefficient ε is given to it, another feature ϕ' becomes more correlated than ϕ_1 to the new residual, thus requiring activation. With both features activated, the optimal coefficient has a negative sign for ϕ_1 , which is thus immediately inactivated, and the same phenomenon is repeated, continuously activating a different feature. The resulting regularization path is illustrated in figure 3.5.

discretization that is dense enough leaves a large number of features, that is exponential in the dimension of the feature space. For example, if the original space \mathcal{X} is \mathbb{R}^m and the feature space consists in Gaussian functions with free centres and free diagonal covariance matrix, there are $2m$ feature parameters, and a discretization of 100 points per parameter gives a set of 100^{2m} features, e.g. 10^{40} for $m = 10$. Hence this sole discretization does not allow to apply directly the exact algorithms on finite feature sets, and an adaptive, approximate optimization procedure is needed.

The DIRECT algorithm introduced in (Jones et al., 1993) is such a procedure that both adaptively samples the space and restricts to a given depth of grid discretization. It simply relies on the function being Lipschitz, without assuming a specific Lipschitz constant, but rather adapting to the local constants (by bounding them). Unlike gradient-based algorithm, it does not spend time on finely improving local optima, which is not useful in the present case, thus providing a good compromise between global and local search. Moreover, it does rely on a general smoothness of the optimized function, but not on its differentiability. The two functions optimized by the homotopy and ASD algorithms are typically continuous in the dictionary's parameters, and differentiable everywhere but at points where the relevant sign for a feature is changing. For example, the ASD is searching for the (approximately) largest *absolute value* of correlation, hence non-differentiability occurs when the sign of the correlation changes. The sketch of the DIRECT algorithm is to start by sampling on a very sparse grid

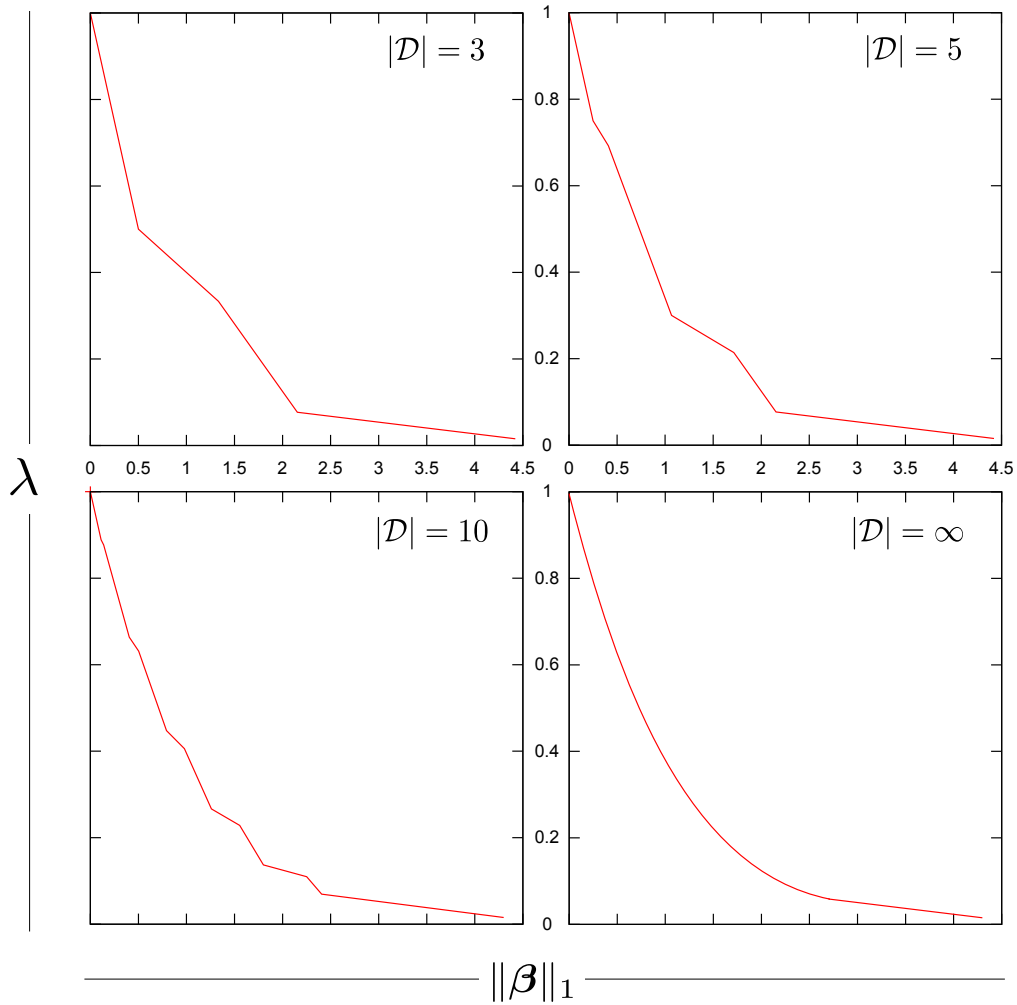


Figure 3.5: Regularization paths of the problem described in section 3.2.1.5 and figure 3.4, for different discretization levels, and for the whole continuous set of features.

The plot is the ℓ^1 -norm of the coefficient against λ , which is linear in between changes in the active set. All features are successively activated, up to a certain point where the two last activated ones remain active until the unregularized least-squares.

and to refine this grid adaptively, on the most promising regions, based on a ratio of the observed values and the size of the gaps, which estimates the local Lipschitz constant. There are only a few parameters to this algorithm: the maximum level of discretization, and the maximum number of samples. The former is not much sensitive, and restricted to powers of three, hence a value of 81 or 243 per dimension is generally a correct choice. The choice of the latter determines a compromise between the execution time and the chances to reach the global optimum, or to reach a sufficiently high value. The optimal choice depends on the complexity of the function, and the consequences of a loose choice depend on the general algorithm. It is more crucial for the homotopy method to reach the global optimum with high probability; otherwise the corrections by adaptive penalizations may produce a consequent drift from the original LASSO problem, and the corrections by active set descent may become prominent, and the actual homotopy steps, based on wrong values, may not make sense anymore, compared to a simple, straight ASD algorithm. As pointed out previously, the precision of the optimization is less important for the ASD method, of which the principle is to activate *an* over-correlated feature, until there is none. The order of activations can influence the number of steps, but it should be noted that this influence is noticeable mostly when a locally suboptimal feature is chosen. Indeed, it can be intuitively understood, especially in the case of radial features, that the set of features to be activated consists in relatively independent features, each contributing to different local parts of the estimator. Activating one does not greatly influence the need for another one, hence the order does not matter much in between those. On the contrary, the neighbourhood of these features consists in similar features that are also over-correlated, and the activation of the “right” – locally most correlated – feature extinguishes the overcorrelation of these neighbours. A wrong local choice leaves the correct feature over-correlated and necessitates additional correction steps when the latter is detected. The DIRECT procedure typically returns a point that is locally optimal, if not globally. This is due to the fact that a region is unexplored – possibly wrongfully – only if the observed values are lower than that observed in other regions, thus the returned point is the result of a deep local optimization.

Another possible optimizing procedure, that shares the same general properties adapted to the problem, consists in performing gradient descents with multiple starting points. The unnecessary of fine local optimization can be taken into account by limiting the degree of precision at this level, multiplying starting points rather than spending efforts on precise local search. However, this limitation only makes sense if it corresponds to a restriction to a predefined grid: it is not harmful to *consistently* ignore the very local neighbourhood of a point, but a simple early stopping of a gradient descent does not prevent subsequent descents to reach a better neighbouring point and induce a sequence of unuseful inactivations and activations. Thus the descents should go through points of the grid and stop when the immediate neighbours on this grid have a lower value. The benefit of this approach is that both optimized functions (correlation and “next-

activation”) are differentiable (by parts), which does give a useful information on where (in which direction) to sample next, whereas DIRECT simply refines the grid (in all or arbitrary directions) around promising points. The fact that differentiability is broken where the most relevant sign of a feature changes does not cause problems, because this occurs where the optimized function is zeroed, i.e. at its minimum value, whence the gradients necessarily yields directions that are opposite to these points. The drawback is that the procedure is not as adaptive as DIRECT: the parallel or successive gradient descents are all fully performed, with no concentration of resources in the most attractive regions.

Both of these optimization methods are rather standard. Existing implementations of DIRECT can be used, and gradient descent methods can be easily implemented or modified to be based on a preliminary grid discretization. However, it can be useful to adapt them further to the present problem, that has the specificity of needing a sequence of optimizations, performed on a sequence of functions that are related to each other. As mentioned above, the activation of a feature does not affect much the correlation or closeness-to-activation of other features that are weakly correlated to that feature. Hence the next optimized function shares a noticeable resemblance to its predecessor, and the optimization should take this into account. In the case of DIRECT, a good and simple enough compromise between re-using the samples locations highly fitted to the previous function, and restarting the procedure from scratch, is to start from an early stage of the previous optimization, where for example one third of the points were sampled, update the values of the new function at these points, and resume the adaptive sampling from there. In the case of multi-start gradient descents, a natural use of the previous optimization for the following one consists in adapting the starting points. They can be set to the points to which they converged in the previous steps, and when several of them had converged to the same point, they can be reallocated to regions in need for exploration. Exact ways of choosing these re-allocations, that are both fruitful and simple, have not been investigated but can certainly be found.

In order to illustrate the relevance of optimization methods given the properties of the optimized functions, and to compare the homotopy and ASD methods on this point, it can be useful to visualize these functions on the illustrative sinusoidal problem used throughout this chapter. The evolution of these functions throughout the first activations is plotted in figure 3.6. The most noticeable property, apart from those illustrating the above considerations, is that the next-activation function optimized by the homotopy is peaky around active features. This function, that we recall here

$$\frac{\phi^T \mathbf{r}^*}{\text{sign}(\phi^T \mathbf{r}^*)w(\phi) - \phi^T \Delta_{\mathbf{r}}}$$

is actually undefined for activated features, since by definition, their correlation with the restricted-least-squares residual (numerator) is zero, its sign is, strictly speaking, undefined, but can be replaced by $s = \text{sign}(\phi^T \mathbf{r})$ and the denominator

is then also zero, from the correlation with the LASSO residual being equal to $s\lambda w(\phi)$. By definition of this function, it can be considered to be equal to λ for active features, but there is a break in the continuity of the function at such a singular point. Both the denominator and numerator tend to zero when approaching it, which translates, as is usually the case, in the ratio tending to different values depending on the “direction” of approach, as can be visualized in figure 3.7 where a zoom is made on an active feature. The first potential resulting problem is that a ratio of quantities tending to zero, in addition to the versatility of the ratio’s limit, can cause computational-precision issues. The second problem is that peaky shapes may be neglected by the optimizing procedure, although they can contain the optimizer. In contrast with this, the simple absolute correlation function optimized by ASD evolves in a stable manner and does not present computational singularities, which is an other strong argument in favour of ASD in the very-large-dictionary setting.

In (Loth et al., 2009), we published the results of experiments on classical regression and classification datasets, and compared with state-of-the-art methods. The method that was used was ECON, that is the homotopy applied to a parameterized space of Gaussian features, the parameters being the centres of the Gaussian and the elements of a diagonal covariance matrix. The DIRECT algorithm was used for the optimization. Small-bandwidth penalization was used, but with a penalization function that later appeared to be less profitable than those mentioned here: rather than normalizing with respect to the values on the observations, a “theoretical” normalization was used, dividing by σ , and, most importantly, the features were then penalized by $1/\sqrt{\sigma}$ (resulting in a normalization/penalization function $w(\phi) = \sqrt{\sigma}$). This penalization function apparently gives almost the same results as a simple normalization, without the benefits of the σ and $\log(1 + \sigma)$ versions discussed here. Nevertheless, the results were very satisfying: often better, and always consistent with the best competing methods. The detailed settings and results are given in 3.5.1.

3.3 Sequential Learning

The algorithms presented in this thesis are designed for the problem setting exposed in the first chapter: one LASSO problem defined by a given set of input/output pairs. This setting where the whole set of data is available at once to the learning algorithm is referred to as *batch learning*. However, different settings may apply, out of necessity or convenience. Data may be collected one by one, or packet by packet, while the estimator is updated to account for the new data. This stream of data may correspond to the evolution of a non-stationary problem. Somewhat similarly, yet differently, the experimenter may also face a sequence of problems that are distinct but related. The first and second settings qualify as *online learning*. When also considering the last setting, we suggest the use of the more general term *sequential learning*. Let us precise in this section

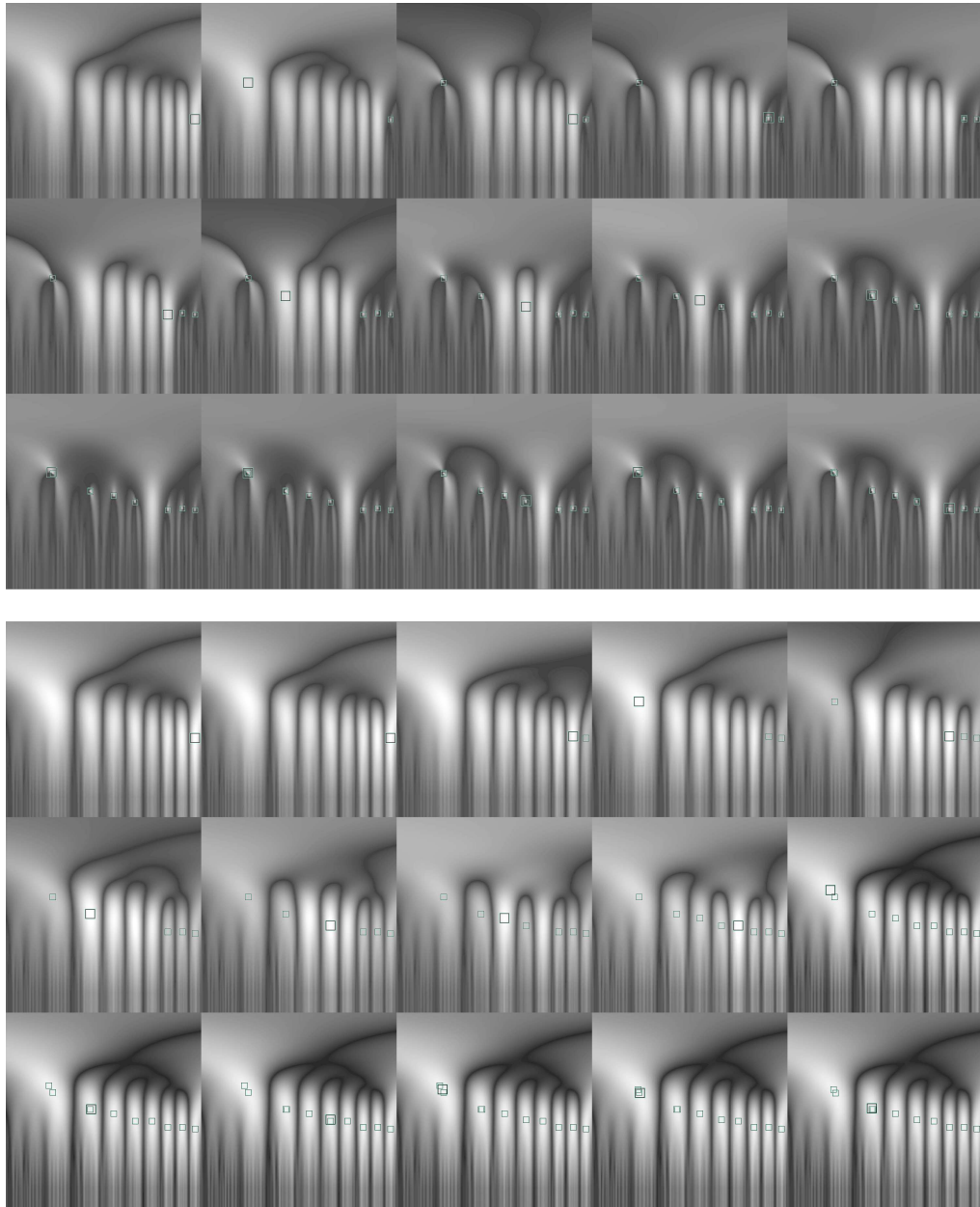


Figure 3.6: Evolution of the function optimized by the homotopy (top: value of λ for which a feature reaches equi-correlation) and ASD (bottom: features correlation), on the sinusoidal function of section 2.6.1.

The two parameters of the features are the centre (x axis) and the bandwidth (y axis). The big square points the optimizer of the function (next feature to be activated), and the smaller squares correspond to already active features. The homotopy functions tend to be peaky around the active features, contrarily to the correlation function of ASD, that keeps a similar shape, which facilitates an approximated optimization.

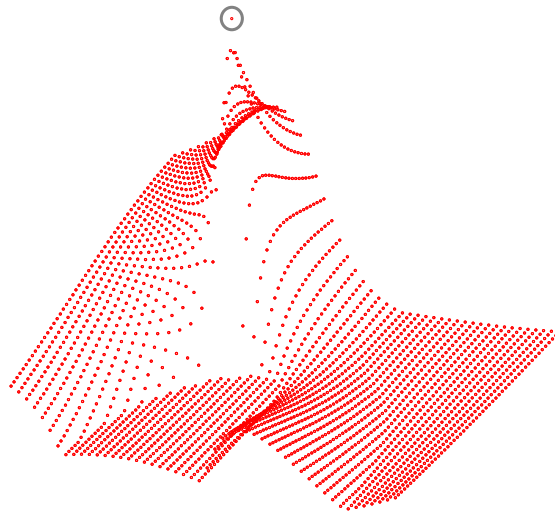


Figure 3.7: An example of the behaviour of the "next-activation" function optimized by the homotopy, in the immediate neighbourhood of an active feature (circled point).

the motivations and forms of these settings, and how well and in which manner the LASSO problem and the three algorithms can be adapted to these settings.

3.3.1 Online Learning

There are several motivations for processing the data in an online fashion. The first is when they are actually collected one by one, for example day by day, yet one needs an estimator before obtaining a complete data set. This estimator should then be updated along, to account for the new data. There may not even be such a thing as a *complete* data set, meaning they simply form a never-ending, or "undefined-ending" stream.

Another motivation is when one can proceed to an *active learning* process. Such a process is defined by the possibility of choosing the samples, or influencing them. A typical example consists in deterministically choosing a point \mathbf{x} of \mathcal{X} and getting a sample of $Y|X = \mathbf{x}$. The main source of information for the choice of the next sample is, understandably, the estimator built from the previous ones. For example, if this estimator indicates a large variance of f in some regions of \mathcal{X} , it is useful to get more samples in these regions. Such an indication is best given by the estimator than by the raw samples, because these samples of (X, Y) are affected by the variance of these variables (noise), which the estimator –assuming it is correct– eliminates ($\hat{f}(\mathbf{x}) \approx f(\mathbf{x}) = \mathbb{E}(\cdot|Y|X = \mathbf{x})$).

It can also be *chosen* to process a given data set in an online fashion, so as to simplify and/or speed up the computation of the estimator. A typical example

is the case of artificial neural networks (ANN) estimators ((Hastie et al., 2008), chapter 11), for which it is impractical to minimize a loss function in an exact, straightforward way, but on which following and propagating partial gradient information for each sample gives good results in reasonable computation time, possibly by processing each sample several times.

Let us explain, regardless of the underlying motivation, how this computation can be performed online, in the case of a simple linear least-squares estimator. It is possible to proceed in the same way as for artificial neural networks, that is by considering the loss $y - \hat{f}(\mathbf{x})$ on the single, current observation, and updating the coefficient vector $\boldsymbol{\beta}$ proportionally to the gradient wrt. this loss, with a small and decreasing factor. This procedure converges to the least-squares solution asymptotically, as one repeatedly process each sample. However, we will not focus on that procedure that does not present as great computational advantages as in the ANN case. The other option is to compute and maintain the exact least-squares solution associated to the observed samples. This procedure is given in chapter 27 of (Lawson and Hanson, 1974), under the name of *sequential accumulation*, for the case where the QR decomposition is used to solve the system. It is better known under the name of *recursive least-squares*. Let us explain here the online procedure in general and its application with the use of the Cholesky decomposition. The least-squares solution is $(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{y}$, in which

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \phi(\mathbf{x}_1) & \dots & \phi(\mathbf{x}_n) \\ | & & | \end{pmatrix} = \begin{pmatrix} - & \phi_1(\mathbf{x}) & - \\ & \vdots & \\ - & \phi_p(\mathbf{x}) & - \end{pmatrix}, \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_p \end{pmatrix},$$

(rows of \mathbf{X} correspond to features and columns correspond to observations), and the modification introduced by a new sample \mathbf{x}_{n+1} is to add the column $\phi(\mathbf{x}_{n+1})$ to \mathbf{X} and the vector $y_{n+1}\phi(\mathbf{x}_{n+1})$ to $\mathbf{X}\mathbf{y}$. The Gram matrix $\mathbf{X}\mathbf{X}^T$ is thus modified by the addition of the matrix $\phi(\mathbf{x}_{n+1})\phi(\mathbf{x}_{n+1})^T$, which is called a rank-one update, and it was explained, in chapter 2, how to consequently update the Cholesky decomposition, and therefore the least-squares solution.

The first important difference between the LASSO and simple least-squares when online learning is involved is that the former uses a regularization parameter λ that needs to be tuned afterwards. This is done by computing the regularization path (the exact one for homotopy, or solutions for different values of λ), and selecting one value from rules or tests like the number of activated features, cross-validation, or C_p Mallow test. It is naturally not practical to recompute this path or set of solutions with each new observation or packet of observations. A smarter way of proceeding is to start with a few first observations and a mid-range value of λ , loosely optimized from these few data, and update the solution given the new observations while questioning from time to time the value of λ , by going back and forth on the regularization path. Thus the LASSO problem and its solution are repeatedly slightly modified by both given new observations,

and chosen drifts of λ in order to optimize it on-the-go.

We have seen that the ASD and homotopy methods also rely on the implicit inversion of $\mathbf{X}\mathbf{X}^\top$ – through its decomposition – with the difference that \mathbf{X} is composed by the *active* features. If one of these algorithms has computed the LASSO solution for $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ and a certain λ , and a new sample $(\mathbf{x}_{n+1}, y_{n+1})$ is presented, the solution can be updated easily if this additional sample does not change the active set σ , since this solution is

$$\boldsymbol{\beta} = (\mathbf{X}_\sigma \mathbf{X}_\sigma^\top)^{-1} (\mathbf{X}_\sigma \mathbf{y} - \lambda \mathbf{w}_\sigma),$$

which presents no other difference to least-squares than the regularization term $\lambda \mathbf{w}_\sigma$. However, the active set is naturally bound to change throughout the sequence of samples. As seen previously, any such change – unoptimality of the active set – is characterized by either the unfeasibility (sign noncompliance) of the new solution, or the overcorrelation of an inactive feature to the new residual. In that situation, the homotopy method is stucked because it relies on following the exact path of solutions, and more precisely on σ being the correct active set for the current λ . The resulting situation when a new sample discards the active set is identical to that when a feature was forgotten due to an imperfect optimization. Using a re-weighting of penalizations is not advisable here, since this partly amounts to modifying the problem to fit the solution, which would tend in this case to bias the problem towards the first observed samples. Moreover, contrary to the case of missed features, new samples can also require some feature inactivations, in which case an ASD-like procedure is needed. Thus, an online homotopy method requires corrections by ASD to track the evolution of the active set through the stream of samples.

The active set descent method adapts more easily to an online setting, since its principle is to correct one by one the unnecessities or insufficiencies of the active set by successive activations and inactivations. The origin of the active set's unoptimality – a more or less informed starting point, a change in λ , or new samples – does not matter in the resolution of this unoptimality. The same argument applies to the cyclical coordinate descent, which constantly and repeatedly adjusts the coefficients to each feature, regardless of the origin of their difference to the solution.

Having recalled that, for the ASD and homotopy method as well as for the simple least-squares, the observed data are essentially compiled into the Gram matrix $\mathbf{X}\mathbf{X}^\top$ – or $\mathbf{X}_\sigma \mathbf{X}_\sigma^\top$ – and its decomposition, one should point out an additional motivation for processing the data online. For the simple least-squares, if the number of observations n is largely greater than the number of features p , the $p \times p$ Gram matrix requires far less storage space than \mathbf{X} itself, and one can even process a potentially infinite stream of observations while storing only this matrix of fixed size, through successive rank-one updates. This also applies to the LASSO and ASD/homotopy case, with the difference that it is not sufficient to store the Gram matrix of *active* features. Indeed, in the form presented in previous chapter, these algorithms need to store at least the whole vector \mathbf{x} itself,

in order to monitor the evolution of the inactive features' correlations. However, if n is significantly greater than the total number of features (and not only active ones), it is possible and useful in term of space to actually maintain the full Gram matrix $\mathbf{X}_{\mathcal{D}}\mathbf{X}_{\mathcal{D}}^{\top}$, of which the entries are the correlations of each feature to each other, which is a sufficient information for the algorithms. The active set's Gram matrix is simply an extract of this full matrix, taking the rows and columns corresponding to the active features. Its Cholesky decomposition, however, is not such a direct extract, yet the update of this decomposition when a feature is added requires only the correlations between the active features and the new one (denoted by $\mathbf{X}\mathbf{u}$ in eq. 2.34, which can be found in $\mathbf{X}_{\mathcal{D}}\mathbf{X}_{\mathcal{D}}^{\top}$).

3.3.2 Non-stationary problems

One special case of online learning is when the stream of data corresponds to a smooth evolution of the random variable (X, Y) (i.e. of the function $y = f(x)$). These non-stationary variables occur for example in economic time series, like exchange rates of currencies. This is also a typical example of genuinely on-line data, with no definite end, and from which an updated estimator is needed throughout the series. Assuming that the evolution of the variables is smooth, that is no abrupt changes occur, one way of handling such data is to consider that recent observations are more relevant for the current estimator than older ones, this relevance evolving as smoothly as the relations between the variables. This can be translated by weighting the observations in the loss function as a function of their recentness:

$$\mathcal{L}\left(\hat{f}, (x_i, y_i)_{i=1, \dots, n}\right) = \sum_{i=1}^n \left(r_{n-i}(y_i - \hat{f}(x_i))\right)^2$$

i.e.

$$\mathcal{L}(\hat{f}, \mathbf{X}, \mathbf{y}) = \|\mathbf{R}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^{\top})\|_2^2$$

where \mathbf{R} is diagonal with $R_{ii} = r_{n-i}$

where r is a relevance function decreasing with the ancientness $n - i$. The Gram matrix $\mathbf{X}\mathbf{X}^{\top}$ is thus replaced by $\mathbf{X}\mathbf{R}\mathbf{X}^{\top}$.

The most common use of such *weighted least-squares* loss functions is to account for heteroskedasticity in (X, Y) ($\text{Var}Y|X$ is not constant with respect to X), following the results of (White, 1980). A natural and practical relevance function consists in an exponential forgetting:

$$r_{n-i} = \gamma^{n-i}, \quad \gamma \in [0, 1)$$

Detailed explanations can be found in, e.g. (Hayes, 1996), but the principle is simple: each new observation is added with a relevance of 1 while the relevance of all previous observations is multiplied by γ . The consequence on the Gram matrix is simple: it is also multiplied by the scalar γ , and the procedure is thus

the same as for the simple recursive least-squares explained in section 3.3.1, with the only difference that the Gram matrix is multiplied by γ before the rank-one update. Therefore, LASSO penalty and ASD or CCD algorithms can be applied without problems, if the full Gram matrix can be stored.

3.3.3 Reinforcement Learning

Reinforcement Learning (RL) is a paradigm in which the problems to be solved can be thought of as sequential regression problems. The precise concepts and notations can be found in e.g. , (Szepesvári, 2010) or in the publications referenced in this section. To summarize them, an *agent* is considered, in an *environment* with which it interacts by selecting actions at each time step: actions affect the *state* of the environment. The *policy* of the agent defines the choice of action given the state of the environment. The learning problem is defined by associating a *reward* to every state-action pair, and setting the goal of maximizing the expected sum of rewards (optimal policy).

The most common approach is to manipulate and estimate these sum of rewards by learning *value functions*: functions that, given a policy, associate to each state (or state-action) the expected sum of rewards to be received when following this policy, starting from this state. In this approach, two categories of algorithms can be identified, that form two radical ways of proceeding, intermediate schemes being possible:

policy iteration the value function of a fixed policy is learnt, then the policy is improved thanks to the knowledge of its value, these two steps being iterated until convergence to the optimal policy;

value iteration improvement and evaluation are conducted simultaneously, state by state; for each considered state, the policy is locally improved (possibly an expected improvement), and an optimistic estimate of the current value function is updated; this corresponds to a generalization of *dynamic programming* (Bellman, 1957).

The evaluation of a policy, given a sufficient number of state / action / reward / next state samples, can be formulated as a regression problem, the response being the rewards, and the estimators being the difference between the value of the next state and the value of the state. If a linear estimator is computed for the value function, the corresponding estimator of the expected reward is also linear, and these estimators can be deduce from each other by this difference or addition. This regression formulation for various settings and algorithms can be found in (Loth and Preux, 2007). All forms of regression methods can thus be applied in the policy iteration framework. However, a specificity of this framework is that the sequence of regressions problems are closely related, since the improvement steps do not greatly affect the value function. Therefore, it can be profitable not to learn each new policy's value from scratch. In this respect, we have

suggested in (Loth et al., 2007) a method for building successive sparse estimators of the value functions, based on the LASSO. It uses an intermediate scheme between policy iteration and value iteration, that consists in collecting a small number of samples of an improved policy, along a *trajectory* (real-life or simulated interactions), and update the value function on these points. For this, the new estimator is defined as the sum of the previous one and a corrective term, and only this corrective term is to be learnt from the new samples. If a LASSO estimator is used, each corrective term is based on a relevant selection of features, but these selections are performed independently, and the selected features add up in the global estimator of the value function, throughout the successive steps that may be numerous: no global, integrated feature selection is performed. The solution that we proposed to this limitation is to account for the previous selections through the penalty factors w : all features that appear in the current estimator of the value function are favoured in the corrective estimator, by being given a lower penalty factor (a fixed fraction of its original penalty). Thus, a compromise is automatically made between fitting the corrections and re-using already selected features: a feature is added only when sufficiently necessary. An other, related, modification is that, in the regression for the corrective term, when the global coefficient of a feature (previous estimator plus correction) is zeroed, it is inactivated, in both the value function estimator and the LASSO algorithm of the correction (homotopy). This results in a sequence of estimators that, from empirical evidences, correctly converge to the optimal value function, and have a sub-logarithmic rate of feature additions: the number of features stabilizes as the global estimator improves.

3.4 Alternative loss functions

Throughout this thesis, we have been focusing on the original and simplest form of ℓ^1 -penalized regression: the classical squared-residual minimization, penalized by the ℓ^1 -norm of the coefficient. The first part is the common basis for regression, due to both its computational simplicity and theoretical justification under i.i.d. Normal noise assumption. The ℓ^1 penalization, as we have seen, is the closest approximation to best-subset selection (ℓ^0 penalization) that offers computational advantages through its convexity. Moreover, it exhibits better properties in terms of generalization (or model recovering) than ℓ^0 penalization itself.

However, different forms of ℓ^1 -penalized regression have been proposed and used, with two types of motivations: robustness, and the specific natures of some problems. Robustness is needed against two forms of disturbances: significantly non-Gaussian and/or non i.i.d. noise, and outliers, that is samples that fall far out of the model, due to either errors in the sampling process or “unlucky” noise. Specific properties of the model and estimator can arise in tasks like signal processing, analysis of grouped variables, or classification.

These different forms of regression affect either the residual-minimizing or coefficient-penalization parts. This section considers some of the most common of these alternatives, and how the ASD, homotopy, and CCD methods can be used or adapted in these contexts. Let us, preliminarily, summarize the requirements of the three methods regarding the loss function:

- as discussed in chapter 2, the requirement for ASD is the convexity of the optimized function, the ability to compute solutions under active constraints (inactive features), and the ability of assessing the benefit of inactivating a constraint (activate a feature); the activation of constraints is monitored by deliberately moving linearly from the minimizer on one active set to that on the next active set, and tracking the first violation;
- the homotopy methods starts on an actual solution of the problem for some value of a regularization parameter λ , and tracks the first of either a constraint violation or a benefit (hence necessity) of a constraint inactivation, as λ decreases (or increases); thus it requires, for computational ease, that the solution evolves piecewise linearly with λ ; such piecewise linear regularization paths were characterized in (Rosset and Zhu, 2007), giving a clear view of which problems can be handled by the homotopy method;
- the convergence of coordinate descent methods requires the separability of the constraint/regularization, that is its being expressed as a sum of independent penalties on each variable (see e.g. (Luo and Tseng, 1992)).

3.4.1 Alternative penalties

3.4.1.1 Elastic net

Least-squares regression is sensible to outliers. Even if the i.i.d. Normal noise assumption is verified, a single observation that happens to present a sensibly larger noise will bias the estimator, if not counter-balanced with a sufficient number of “normal” observations. The additional ℓ^1 penalty does not especially correct such drawbacks. On the contrary, in the presence of “local” features that may fit an outlier without affecting other points, and in the absence of a large-bandwidth preference mechanism as exposed in section 3.2.1.3, such an overfitting feature can be selected preferably to model-fitting ones. A classical way to encourage smoothness of the estimators and thus lessen the effect of outliers is to penalize the ℓ^2 -norm of the coefficient, which associated to least-squares is often called *ridge regression*, from (Hoerl and Kennard, 1970b):

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \|\mathbf{y} - \mathbf{X}^T \boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2 \quad (3.27)$$

The idea of *elastic net*, as proposed in (Zou and Hastie, 2005), is to add both

ℓ^2 and ℓ^1 penalty terms to the squared-residual loss:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \|\mathbf{y} - \mathbf{X}^T \boldsymbol{\beta}\|_2^2 + \lambda_2 \|\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 \quad (3.28)$$

The two quadratic terms of (3.27) easily combine together, and simple derivation gives as the unique solution

$$\boldsymbol{\beta}_{\text{ridge}} = (\mathbf{X}\mathbf{X}^T + \lambda_2 \mathbf{I})^{-1} \mathbf{X}\mathbf{y} \quad (3.29)$$

It is proposed in (Zou and Hastie, 2005) to rewrite (3.28) as an equivalent LASSO problem by the mean of p dummy additional observations, so as to solve it by directly applying the LAR-LASSO (i.e. homotopy) algorithm, for a fixed set of values for λ_2 . However, this reformulation of ridge regression, as a straight least-squares problem on an augmented observation set, and the equivalent reformulation of elastic net as LASSO, though enlightening, is not practical as it adds unnecessary complexity to the solving algorithm. The authors suggest to correct this by low-level adjustments taking into account the sparsity of the augmented observations. We advocate that, should an adaptation of the LASSO algorithm be made, a simpler solution, resulting in essentially the same algorithm, consists in simply replacing the least-squares loss by the ridge loss in the derivations of chapter 2, which ultimately results in replacing the Gram matrix $\mathbf{X}\mathbf{X}^T$ by $\mathbf{X}\mathbf{X}^T + \lambda_2 \mathbf{I}$, $\boldsymbol{\beta}^*$ becoming the restricted ridge estimator rather than the restricted least-squares estimator. As mentioned in (Zou and Hastie, 2005), the change induced in the updates of the Cholesky decomposition is minimal: when adding a feature, (2.35) is replaced by

$$w = \sqrt{\mathbf{u}^T \mathbf{u} + \lambda_2 - \mathbf{v}^T \mathbf{v}}$$

Thus, both the homotopy and ASD easily adapt to the elastic net. However, ASD presents once again an advantage: using the homotopy requires to solve successive λ_1 -regularization paths corresponding to a sequence of values of λ_2 . With ASD, it is possible, at any step, to change either λ_1 or λ_2 (or both), thus one can, for example, compute the λ_1 -path for a value of λ_2 , retain the best value of λ_1 from validation tests and compute the λ_2 -path for this value, and iterate then process.

The CCD algorithm naturally also adapts easily to the elastic net.

3.4.1.2 Group Lasso

It is sometimes desirable to select groups of variables rather than individual variables, or to evaluate the importance of each group in the model. This motivation lead to the formulation of the *group LASSO* in (Bakin, 1999), and further developments in (Yuan and Lin, 2006). The p features of the dictionary are partitioned in l groups, and rather than the sum of absolute values of the coefficients, the

sum of ℓ^2 -norms of the groups coefficients is used as a penalty term :

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \frac{1}{2} \|\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}\|_2^2 + \lambda \sum_{i=1}^l \sqrt{p_i} \|\boldsymbol{\beta}_i\|_2 \quad (3.30)$$

where $\boldsymbol{\beta}_i$ is the vector of coefficients of the i -th group of features, and p_i is the cardinal of group i . The penalty term introduced in trove.nla.gov.au/work/34383173 was, for each group i , $\sqrt{\boldsymbol{\beta}_i^\top \mathbf{K}_i \boldsymbol{\beta}_i}$. The particular case retained here as well as in (Yuan and Lin, 2006), corresponds to $\mathbf{K}_i = p_i \mathbf{I}$. p_i plays the same role as the penalization factors w in our definition of the LASSO: assuming the features are normalized, values equal to the cardinals of the groups “normalizes” the groups themselves by cancelling the effects of their respective sizes on selection. Different values allow to deliberately favourize or penalize certain groups. Let us assume that all features are preliminary scaled so as to ensure this fair or biased group selection, leaving the following simpler form of (3.30):

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \frac{1}{2} \|\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}\|_2^2 + \lambda \sum_{i=1}^l \|\boldsymbol{\beta}_i\|_2 \quad (3.31)$$

We can also assume, without loss of generality, that the features *are* normalized and the penalty factors equal to one.

Denoting by \mathbf{X}_i the sub-matrix of \mathbf{X} corresponding to features of group i , the KKT conditions translate in

$$\begin{cases} \mathbf{X}_i(\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}) = \lambda \frac{\boldsymbol{\beta}_i}{\|\boldsymbol{\beta}_i\|_2} & \text{for active groups,} \\ |\mathbf{X}_i(\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta})| \leq \lambda & \text{for inactive groups,} \end{cases} \quad (3.32)$$

which give the necessity and sufficiency conditions to update the active set in ASD, homotopy, and CCD. When assuming (ASD), or knowing (homotopy) an optimal active set σ of groups, the remaining of the ℓ^2 -norm of group coefficients does not permit a simple expression of the solution from (3.32). A first step of simplification is to solve by blockwise coordinate descent: solve for one group while fixing the others' coefficients, and iterate. i being the group to solve for, $\boldsymbol{\beta}_i$ verifies

$$\mathbf{X}_i(\mathbf{y} - \mathbf{X}_{\sigma-i}^\top \boldsymbol{\beta}_{\sigma-i}) = \lambda \frac{\boldsymbol{\beta}_i}{\|\boldsymbol{\beta}_i\|_2}, \quad (3.33)$$

hence

$$\mathbf{X}_i(\mathbf{y} - \mathbf{X}_{\sigma-i}^\top \boldsymbol{\beta}_{\sigma-i}) - \mathbf{X}_i^\top \boldsymbol{\beta}_i = \lambda \frac{\boldsymbol{\beta}_i}{\|\boldsymbol{\beta}_i\|_2}, \quad (3.34)$$

and

$$\boldsymbol{\beta}_i = (\mathbf{X}_i \mathbf{X}_i^\top + \frac{\lambda}{\|\boldsymbol{\beta}_i\|_2} \mathbf{I})^{-1} \mathbf{X}_i(\mathbf{y} - \mathbf{X}_{\sigma-i}^\top \boldsymbol{\beta}_{\sigma-i}) \quad (3.35)$$

which corresponds to a ridge regression solution, except for the interdependency between the parameter and the norm of the solution. The second step of simplification taken in (Yuan and Lin, 2006) consists in imposing orthonormality between the features of each group: for each pair of features ϕ_1, ϕ_2 in the same group,

$$\begin{aligned}\phi_1^\top \phi_2 &= 0 \\ \phi_1^\top \phi_1 &= \phi_2^\top \phi_2 = 1\end{aligned}$$

This is of course a strong assumption, but it simplifies (3.35) to

$$\boldsymbol{\beta}_i = (\mathbf{I} + \frac{\lambda}{\|\boldsymbol{\beta}_i\|_2} \mathbf{I})^{-1} \mathbf{c}_i \quad (3.36)$$

$$= \frac{\|\boldsymbol{\beta}_i\|_2}{\|\boldsymbol{\beta}_i\|_2 + \lambda} \mathbf{c}_i, \quad (3.37)$$

with $\mathbf{c}_i = \mathbf{X}_i(\mathbf{y} - \mathbf{X}_{\sigma-i}^\top \boldsymbol{\beta}_{\sigma-i})$ denoting the correlation vector of features from group i to the residual from other active groups. The solution is then

$$\boldsymbol{\beta}_i = (1 - \frac{\lambda}{\|\mathbf{c}_i\|_2}) \mathbf{c}_i,$$

since it implies

$$\begin{aligned}\|\boldsymbol{\beta}_i\|_2 &= (1 - \frac{\lambda}{\|\mathbf{c}_i\|_2}) \|\mathbf{c}_i\|_2 \\ &= \|\mathbf{c}_i\|_2 - \lambda\end{aligned}$$

and

$$\mathbf{c}_i = \frac{\|\mathbf{c}_i\|_2}{\|\mathbf{c}_i\|_2 - \lambda} \|\boldsymbol{\beta}_i\|_2$$

hence

$$\begin{aligned}\frac{\|\boldsymbol{\beta}_i\|_2}{\|\boldsymbol{\beta}_i\|_2 + \lambda} \mathbf{c}_i &= \frac{\|\mathbf{c}_i\|_2 - \lambda}{\|\mathbf{c}_i\|_2} \mathbf{c}_i \\ &= \frac{\|\mathbf{c}_i\|_2 - \lambda}{\|\mathbf{c}_i\|_2} \frac{\|\mathbf{c}_i\|_2}{\|\mathbf{c}_i\|_2 - \lambda} \|\boldsymbol{\beta}_i\|_2 \\ &= \|\boldsymbol{\beta}_i\|_2\end{aligned}$$

However, the orthonormality restriction is not needed to solve (3.35) with a reasonable complexity. We propose the following method: solve for a “guess” value of $\|\boldsymbol{\beta}_i\|_2$, correct the guess from the result, and iterate until convergence. This is possible from the following property: the ridge parameter $\lambda/\|\boldsymbol{\beta}_i\|_2$, similarly as for the t vs. λ relation in the LASSO, corresponds to a constraint on the ℓ^2 -norm of the solution, and, for the same reasons as in the LASSO, they are

decreasing with respect to one another. Thus if, making a guess z for $\|\boldsymbol{\beta}_i\|_2$, the norm of

$$\hat{\boldsymbol{\beta}}_i = (\mathbf{X}_i \mathbf{X}_i^\top + \frac{\lambda}{z} \mathbf{I})^{-1} \mathbf{c}_i$$

is largest than z , this norm must be lowered, by increasing the regularization parameter $\frac{\lambda}{z}$, i.e. decreasing z . The largest possible value of $\|\boldsymbol{\beta}_i\|_2$ is the unregularized one

$$z_{\max} = \|(\mathbf{X}_i \mathbf{X}_i^\top)^{-1} \mathbf{c}_i\|_2,$$

and successive binary cuts in $[0, z_{\max}]$ yield an exponential rate of convergence to the solution. The updates of the inverse matrix can be done easily and efficiently through its Cholesky decomposition.

The (non-orthonormal) group LASSO can thus be solved by mean of an active set descent that identifies the optimal active set of groups, the inner computations of restricted solutions being conducted by blockwise coordinate descent on the current active set, and the inner computation of the latter relying on the binary cuts described above. The question is opened of whether blockwise coordinate descent is mandatory, or if the ℓ^2 -norms guesses can be conducted concurrently on the whole active group set.

It can also be solved by one global blockwise coordinate descent on all groups, setting $\boldsymbol{\beta}_i = \mathbf{0}$ when $|\mathbf{X}_i(\mathbf{y} - \mathbf{X}_{\sigma-i}^\top \boldsymbol{\beta}_{\sigma-i})| \leq \lambda$, and computing it by the binary search otherwise. The method given in (Yuan and Lin, 2006) is such a blockwise descent, with the simplified computation induced by the orthonormality assumption.

Both the orthonormal and non-orthonormal versions of the problem yield nonlinear regularization paths, hence the homotopy method can not be used. However, (Yuan and Lin, 2006) proposes the *group LARS* algorithm, adapted from the LAR algorithm of (Efron et al., 2004) (equivalent to the homotopy, or LAR-LASSO, without inactivations) that computes estimators that are close to group LASSO estimators, as LAR estimators are close to LASSO estimators.

3.4.2 Alternative residual losses

3.4.2.1 Least Absolute Deviations

The ridge penalization lessens the effects of outliers by encouraging smoothness of the estimator, the coefficients tending to be of the same range. There is a somewhat more direct way of preventing these effects, by using a different loss for the residual itself. The simplest alternative is to minimize the ℓ^1 -norm of the residual rather than its ℓ^2 -norm:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \|\mathbf{y} - \mathbf{X}^\top \boldsymbol{\beta}\|_1 = \sum_{i=1}^n |y_i - \phi(\mathbf{x}_i)^\top \boldsymbol{\beta}|, \quad (3.38)$$

which is known as *least absolute deviations* (LAD) regression. It is a best unbiased linear estimator under the assumption of a Laplacian noise. Figure 3.8 shows the

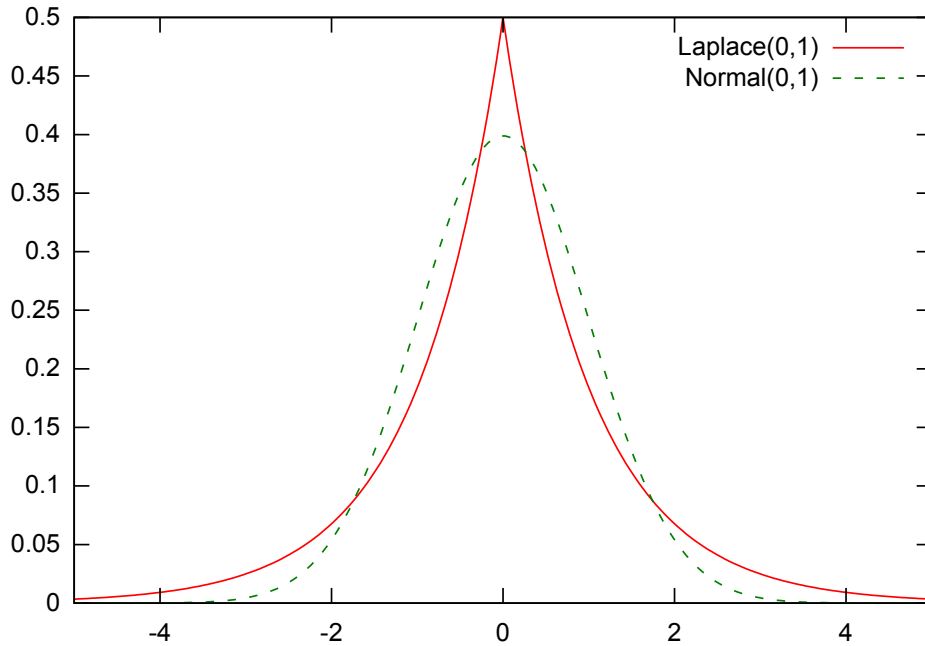


Figure 3.8: Density functions of standard Normal and Laplace distributions, that are the assumed distributions of the noise for, respectively, the least-squares and least-absolute-deviations estimators.

The latter is less influenced by large noise terms, since they are more likely under Laplace distributions.

density functions of the standard Normal and Laplace distributions, which shows that large deviations from the mean is by far more likely in Laplace distributions, and have thus a lower impact when estimating the mean. A LAD estimator can be computed by iteratively re-weighted least-squares (IRLS), or, more efficiently, the problem can be transformed into a linear program (LP), the absolute-value operator being handled by sign monitoring, in the very same way as the regularization term of the LASSO. Similarly to the LASSO, standard LP algorithms, or more dedicated adaptations may then be used. In (Armstrong and Frome, 1976) is given a comparison between the IRLS approach and such an adaptation of the simplex algorithm. We have mentioned in the introduction of section 2.2 that active set methods apply to QP the same principles as the simplex algorithm for LP. Actually, the ASD algorithm as stated in Alg. 1, when applied with a linear function to be minimized, becomes the exact simplex algorithm. The step of finding a minimizer subject to the active constraint is not as well defined as for strictly convex functions, because this minimizer is necessarily asymptotic; nevertheless, the linear move from the current solution to this minimizing direction is well defined, and an inactive constraint is necessarily activated on the way, unless the problem has no solution.

When adding an ℓ^1 regularization term, the problem is still equivalent to a

linear program, and standard LP algorithms as well as adapted ASD/simplex can be used. The regularization path is naturally piecewise linear, thus an homotopy method can be derived from ASD, in the same way the LASSO homotopy was derived from the LASSO ASD in section 2.3. The resulting algorithm has been proposed in (Wang et al., 2006), though not from the same derivations.

The essential difference introduced in ℓ^1 regularization by the use of the LAD loss is that, in addition to monitoring the sign changes in β , the sign changes in the residual vector must also be tracked. One way of seeing things is that each of these sign changes yields a new version of the linear loss function. Therefore, the CCD algorithm does not cope well with ℓ^1 -regularized LAD, since at every change in the coefficients, which are numerous with CCD, these changes in the loss function must be monitored.

3.4.2.2 Huber loss

The same principle apply to the use of a Huber loss function. This loss, introduced in (Huber, 1964), combines the LAD properties for outliers and the least-squares properties for regular samples. It is defined, given a positive parameter δ , by

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \hat{f}) = \sum_{i=1}^n l(x_i, y_i, \hat{f}) \quad (3.39)$$

with

$$l(x, y, \hat{f}) = \begin{cases} \frac{1}{2}(y - \hat{f}(x))^2 & \text{if } |y - \hat{f}(x)| \leq \delta \\ \delta(|y - \hat{f}(x)| - \delta/2) & \text{if } |y - \hat{f}(x)| \geq \delta \end{cases} \quad (3.40)$$

Each time the residual on an observation (x_i, y_i) reaches the $\pm\delta$ barrier, the loss function is changed, similarly to the LAD loss when the sign of this residual's changes. One difference is that the loss function associated to the current active set (with respect to both the features and observations constraints) may be either quadratic or (less likely) linear. Otherwise, the same considerations apply to the Huber and LAD losses regarding the adaptation of ASD, homotopy, and CCD algorithms. The homotopy version is explained in (Rosset and Zhu, 2007), with the mention that the same principles apply to other ℓ^1 -based loss functions (e.g. LAD or SVM's hinge loss).

3.4.2.3 Logistic regression

The logistic regression model is designed for classification tasks, where the outcome Y is one of K classes $1, \dots, K$. It assumes a generative model where the class of a predictor \mathbf{x} is drawn from a multinomial distribution with probabilities $(p_1(\mathbf{x}), \dots, p_K(\mathbf{x}))$, and estimates these probabilities by fitting a linear model to

their *logit*:

$$\text{logit}(p_i) = \log\left(\frac{p_i}{1-p_i}\right)$$

which amounts to estimating p_i by the *logistic* function of a linear expression:

$$\hat{p}_i = \frac{1}{1 + \exp(-\mathbf{X}^\top \boldsymbol{\beta})}$$

Maximizing the log-likelihood of the observations with respect to the coefficient $\boldsymbol{\beta}$ can be done (see for example (Hastie et al., 2008), 4.4) by the method of iteratively re-weighted least-squares.

The IRLS procedure can be used as the inner component of an active set descent when adding an ℓ^1 coefficient penalization, as explained in (Lokhorst, 1999) and (Lokhorst, 1999). These two publications adapt the version of (Osborne et al., 2000b), that is for the constrained formulation of the LASSO. However, the regularized formulation can be handled as easily: as for the LASSO, once an active set is fixed, the ℓ^1 -penalty term becomes a simple linear expression that incorporates without much complication into the (weighted) least-squares, and even more easily than the equivalent ℓ^1 constraint.

The regularization path is nonlinear, whence the homotopy algorithm cannot be applied. An approximated homotopy method has been presented in (Rosset and Zhu, 2007) for regularized logistic regression and other nonlinear-path problems, that consists in approximating the solution path, to identify and converge to the next change in the active set. However, given the similar if not better properties of ASD with respect to the homotopy, and the fact that it does not especially suffer from the non-linearity of the path, ASD seems to be a natural choice among the two in such cases.

If there is not too much correlation between the features, CCD can be a better choice, since the notion of linearity does not appear at a single-coordinate level. (Friedman et al., 2010) extensively presents the application of coordinate descent to regularized generalized linear models, that is linear models transformed by one nonlinear function, as the logistic function for logistic regression.

3.5 ECON Experiments

3.5.1 ECON with homotopy and adaptive penalization

In (Loth et al., 2009), our proposal, under the name of ECON, of using the LASSO with a parametric space of features and using the homotopy method (algorithm 12), was supported by experimental evidences, already mentioned in section 3.2.2. We recall that, in order to cope with features that have become overcorrelated due to the approximate maximization, algorithm 12 consequently adapts their penalizations so that they become equicorrelated. The detailed settings and results of these experiments on this first ECON algorithm are given in the following.

3.5.1.1 Methodology

We compared the performance of ECON with published results on Support Vector Machine (SVM) (Rätsch et al., 2001), Relevance Vector Machine (RVM) (Tipping, 2001), various boosting algorithms investigated by Rätsch (Rätsch et al., 2001) and the Kernel Fisher Discriminant (KFD) (Mika et al., 1999) as reported on the website (Rätsch, web), the kernel basis pursuit (Guigue et al., 2005), and the LOGREG-LASSO algorithm (Roth, 2004). These published results provide an interesting basis of comparison; SVM are well-known to perform extremely well in supervised classification tasks in terms of accuracy, though not being sparse; RVM is meant to be sparser than SVM; boosting algorithms are known to perform very well too for supervised classification; KFD was reported as also very good for classification. In the LARS family which is meant to provide sparse estimators, kernel basis pursuit and LOGREG-LASSO both perform very well on classification, and regression tasks. The repository of (Rätsch, web) provides an interesting base for such experiments, because a fair comparison of algorithms is delicate task. The study that the repository provides has been conducted with care and precision, and we have ourselves taken care of following the exact same settings and measures.

Each dataset was split into a training set and a test-set using the same amount of data in each set as in the cited publications. 100 such splits were performed on each dataset, giving 100 different runs on each dataset. This provides performance on which statistics may be performed. To cope with the published results, we compared the mean-square error; its standard deviation provides a measure of the variability of the precision of predictions.

In the performance reported below on supervised classification, and regression tasks, we performed a fixed amount of iterations (500), and then retained the best MSE measured on the test set during these iterations. The results in tables 3.2 and 3.3 are averaged over the 100 training-testing splits.

3.5.1.2 Supervised classification

ECON is essentially a regression algorithm. To deal with supervised binary classification, we encoded the two classes as ± 1 ; to predict the label of a data, we consider the sign of the prediction for this data: $\text{sign}(\hat{y}(\mathbf{x}))$.

Table 3.2 presents the results: on 5 datasets out of 10, ECON obtains the best results in accuracy; ECON improves the best accuracy by 3.5 % on these 5 datasets. On the other 5 datasets, one (Ringnorm) seems to exhibit an anomaly since there is an order of magnitude between ECON performance and other algorithms; this dataset does have a training set which is much smaller than the test set (15 times smaller), but the situation is the same for the Titanic dataset where ECON performs the best, and Twonorm where the difference in performance is not so large (by far); furthermore, the dimension m of the data is the same for Ringnorm and Twonorm, so that we conclude that this is also not the reason for this order of magnitude. So, let alone this anomalous Ringnorm,

Table 3.2: Comparative results of ECON on classification problems

Dataset	m	best from (Rätsch, web)	RVM	LR-LASSO	ECON
Banana	2	LP _{Reg} -AB: 10.73(0.43)	10.8	10.7(0.5)	10.1 (1.9)
Breast-cancer	9	KFD: 24.77(4.63)	29.9	26.1(4.6)	23.3 (4.3)
Diabetes	8	KFD: 23.21(1.63)	NA	23.5(1.9)	22.7 (1.8)
Flare-solar	9	SVM: 32.43 (1.82)	NA	33.3(1.6)	32.5(1.7)
German	20	SVM: 23.61(2.07)	22.2	23.63(2.3)	23.2 (2.1)
Heart	13	SVM: 15.95 (3.26)	NA	16.0(3.1)	16.2(3.5)
Ringnorm	20	KFD: 1.49(0.12)	NA	1.8(0.3)	10.4(3.0)
Thyroid	5	KFD: 4.2 (2.07)	NA	4.8(2.3)	4.6(2.3)
Titanic	3	SVM: 22.42(1.02)	23.0	22.9(1.2)	22.0 (0.8)
Twonorm	20	KFD: 2.61(0.15)	NA	2.6 (0.2)	2.9(0.4)

on the other 4 datasets on which ECON is not the best, ECON is only beaten by a small factor; on Thyroid, and Twonorm, the performance of ECON is better than that of SVM, and of the same order as boosting (please, refer to (Rätsch, web) for these figures not reported here).

So despite being an algorithm for regression, and definitely not relying on any principle of margin maximization, ECON obtains “state of the art” performances on supervised classification problems. Furthermore, ECON provides sparse estimators. They are much sparser than those provided by SVM, and a bit less sparse than those provided by LOGREG-LASSO, and RVM, but more accurate.

In more details, table 3.2 provides the following information about classification results on UCI benchmark datasets. For each dataset, we provide the number m of attributes of the data, and the results of different algorithms. The column entitled “best from (Rätsch, web)” provides the best results available from Rätsch’s data available on his website (Rätsch, web) and discussed in (Rätsch et al., 2001; Mika et al., 1999); they compared 7 algorithms: RBF-networks with a fixed number of Gaussian units, Adaboost, and variants (LP_{Reg}-AdaBoost, QP_{Reg}-AdaBoost, AdaBoost_{Reg}), SVM, and Kernel Fisher Discriminant; we only provide the best results among these (highly performing) algorithms. SVM uses Gaussian kernels. For each algorithm, we provide a pattern of the form $x(y)$ where x denotes the average MSE, y its standard deviation. The best performance is highlighted with boldface font (there is a peculiarity here that the results provided in (Rätsch, web) are not always coherent with those provided in (Roth, 2004); actually, the results provided here show a better accuracy than the latter reference).

3.5.1.3 Regression problems

For regression problems, we illustrate ECON performance on a few standard benchmark datasets in this section. Again, we use the same datasets as (Roth,

Table 3.3: Comparative results of ECON on regression problems.

DATASET	P	SVM	RVM	LR-LASSO	ECON
FRIEDMAN #1	10	2.92/116.2	2.80/59.4	2.84/73.5	2.16 /18
FRIEDMAN #2	4	4140/110.3	3505 /6.9	3808/14.2	4062.46/10
FRIEDMAN #3	4	0.0202/106.5	0.0164/11.5	0.0192/16.4	0.0139 /24
ABALONE	8	4.37/972	4.35 /12	4.35 /19	4.59/63
BOSTON HOUSING	13	8.04	7.46	NA	11.51/87

2004), using the same training, and testing sets. Some results are presented in table 3.3. We compare the performance of ECON with those published in (Roth, 2004), concerning the Support Vector Machine, the Relevance Vector Machine, and the LOGREG-LASSO. For each algorithm, we provide the accuracy measured on a test set. In the table, boldface font highlights the best performance for each dataset.

We note the very good performance of ECON on Friedman’s F1 and F3 functions. On Abalone, though not the best, ECON performs quite well with regards to other algorithms. On Boston housing, it is a little bit behind SVM and RVM, though still very competitive with most other algorithms and published results.

3.5.1.4 Practical application to photometric solids

In (Loth et al., 2009) are also reported the results of experiments on the practical problem of simulating light sources of different natures in computer graphics. The classical procedure consists in characterizing a light source with numerous measurements sampled in different directions of the light, that form what is denoted as the *photometric solid* of the light source. A linear interpolation is then used for intermediate values in between these samples. As an alternative to this space and time consuming reconstruction of the light source, we have proposed to use a model built by the ECON procedure. ECON was able to provide sparse and accurate models, yielding no distinguishable differences in the resulting scenes (see fig. 3.9), thus providing a more efficient simulation than that using interpolation, and a better accuracy than the best alternative based on artificial neural network models. The details of the experiments can be found in the article.

3.5.2 ECON with ASD

Following the conclusion that ASD is simpler, less perturbed by approximate optimization, and offers a smoother function to optimize than the homotopy method, ECON was re-implemented with that LASSO algorithm. Experiments were run on the same problems as in (Loth et al., 2009).

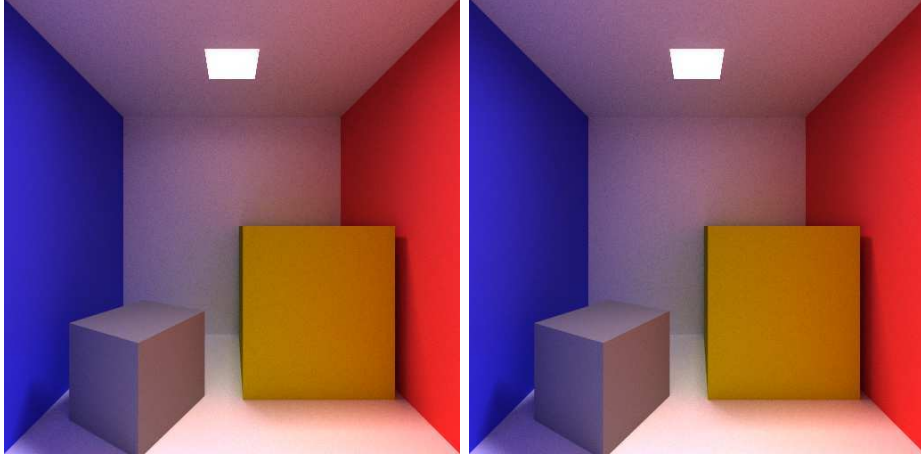


Figure 3.9: Results of a light source simulation, when using interpolation or an ECON model.

No visible difference appears, while the ECON model lessens the space and time complexity of the reconstruction.

One of the benefits was the ability to use perceptron features, that presented serious computational stability issues with the homotopy method. This class of features was defined by the following:

In all experiments, the natural features, that is the original attributes of the data, were used, and one or several feature classes were added to the feature dictionary. For no clear reason, the procedure of normalizing the features ($\forall \phi \in \mathcal{D}, \sum_{\mathbf{x} \in \text{training set}} \phi(\mathbf{x})^2 = 1$) and penalizing them by some measure of their generalization property (e.g. $1/\sigma$ for Gaussian features), did produce hazardeous results, probably linked to arithmetical precision issues. Therefore, the following classes of features were used, simply letting the range of their values affect the LASSO selection (the larger $\sum_{\mathbf{x} \in \text{training set}} \phi(\mathbf{x})^2$, the more likely ϕ is activated):

- **Gaussian-CST** (for *constant*): for a fixed value of σ ,

$$\left\{ \mathbf{x} \mapsto \exp \left(-\frac{\|\bar{\mathbf{x}} - \mathbf{c}\|_2^2}{2\sigma^2} \right) \mid \mathbf{c} \in [0, 1]^d \right\}.$$

- **Gaussian-ISO** (for *isovariant*):

$$\left\{ \mathbf{x} \mapsto \exp \left(-\frac{\|\bar{\mathbf{x}} - \mathbf{c}\|_2^2}{2\sigma^2} \right) \mid \mathbf{c} \in [0, 1]^d, \sigma \in [0, 4.26) \right\}.$$

- **Gaussian-DIAG** (for *diagonal*):

$$\left\{ \mathbf{x} \mapsto \exp \left(\frac{1}{2} \sum_{i=1}^d \left(\frac{\bar{x}_i - c_i}{\sigma_i} \right)^2 \right) \mid \mathbf{c} \in [0, 1]^d, \boldsymbol{\sigma} \in [0, 4.26)^d \right\}.$$

- **perceptron:**

$$\left\{ \mathbf{x} \mapsto 1 - 2 \left(1 + \exp \left(-\frac{\mathbf{w}^\top \bar{\mathbf{x}} + b}{\sigma^2} \right) \right)^{-1} \mid \mathbf{w} \in [-1, 1]^d, b \in [-1, 1], \sigma \in [0, 1] \right\}.$$

- **polynomial:**

$$\left\{ \mathbf{x} \mapsto \frac{1}{p} (\mathbf{w}^\top \bar{\mathbf{x}} + b)^p \mid \mathbf{w} \in [0, 1]^d, b \in [0, 1], p \in [1, 3] \right\}.$$

where d is the dimension of the predictors, $\bar{\mathbf{x}}$ denotes the result of normalization on \mathbf{x} , such that all variables of all training predictors belong to $[0, 1]$.

In all experiments also, results were averaged over 100 different training/test sets (trials). Letting λ_{\max} be the highest correlation to the response \mathbf{y} (corresponding to the first activated feature), an exponentially decreasing sequence of values was defined for λ :

$$\lambda_0 = \lambda_{\max}, \lambda_1 = 0.8 * \lambda_0, \dots, \lambda_{i+1} = 0.8 * \lambda_i, \dots$$

and the least test MSE among this sequence was retained for each trial.

3.5.2.1 Classification

We first repeated the classification experiments with the same features as in (Loth et al., 2009) (in which the same normalization/penalization scheme was used), that is the Gaussian-DIAG class. The results, reported in table 3.4, were consistently better, confirming the adequacy of ASD for handling the infinite feature sets of ECON.

dataset	min MSE
banana	11.0(0.5)
breast-cancer	23.0(4.0)
diabetis	22.0(1.5)
flare-solar	32.2(1.7)
german	22.7(2.0)
heart	14.1(3.0)
ringnorm	2.9(0.3)
thyroid	4.0(1.9)
titanic	21.8(1.3)
twonorm	2.7(0.2)

Table 3.4: mean minimum prediction errors (percentage), and standard deviations, when using ECON with Gaussian-DIAG features and ASD algorithm. The results are consistently better than when using the homotopy algorithm, as in section 3.5.1.2 (table 3.2)

3.5.2.2 Friedman functions

On the Friedman functions, various classes of features were used, in order to assess the benefits of using different classes, that can be more or less rich.

In addition to the linear (original) features, the first experiments used Gaussian features with a common, fixed, bandwidth (Gaussian-CST) of which the centers were either spread among a grid of more or less fine resolution, or using as centers the training predictors (which we refer to as the *kernel* setting), or letting ECON choose them freely. Except for the last setting, the feature dictionary \mathcal{D} was thus finite. The common bandwidth was chosen as the one giving the best results in the kernel setting. The other classes listed in page 160 were then used, as well as the combination of Gaussian-DIAG, perceptron, and polynomial.

The results reported in table 3.5, show that the richness of Gaussian-DIAG, with which the bandwidths are set independently for each dimension of each feature, can be a benefit as for the functions #1 and #3, or a drawback as for the function #2. The first function has five dummy variables that are not part of the model, hence ECON assigns the largest possible bandwidths in these dimensions, thus focusing the estimator on the other variables, and the better results are not surprising. The results on function #2 show that the preference on large-scale features does not systematically prevent the risk of overfitting, as was the case in the sinusoidal example and with the Friedman function #3.

The best results are indeed obtained with different feature classes for the three functions, reproducing the issue of choosing the right kernel in kernel methods. However, with ECON – and LASSO estimators in general – different classes can be used together, as in the last settings of these experiments. This gave results that are always better than the averages of using the classes individually. In the case of function #3, these mixing even gives better results than with each separate class. A good general-purpose feature dictionary seems to be the combination of Gaussian-DIAG and polynomials – and original attributes, which is not surprising since they present very distinct, hence complementary, shapes, and correspond to the two mostly used kernels in kernel methods.

		Friedman #1		Friedman #2		Friedman #3	
		MSE	time	MSE	time	MSE _(10⁻²)	time
	linear	6.11(0.37)	0.0	19785(1165)	0.0	4.18(0.34)	0.0
	σ	1.0		1.7		0.7	
Gaussian CST	grid 2	6.11(0.37)	1.9	1259(440)	0.1	2.70(0.28)	0.1
	grid 3	4.27 ¹	1983	1297(441)	0.5	1.15(0.19)	0.8
	grid 4			1290(440)	1.9	1.15(0.19)	3.3
	grid 5			1286(438)	5.4	1.14(0.19)	9.8
	grid 6			1283(436)	12.5	1.14(0.19)	23.4
	grid 7			1282(435)	25.6	1.13(0.18)	45.0
	grid 8			1280(435)	48.1	1.12(0.18)	81.2
	kernel	2.71(0.27)	5.7	1241(425)	1.2	1.12(0.19)	3.4
	ECON	2.63(0.30)	24.7	1260(427)	7.4	1.13(0.19)	19.4
ECON	Gaussian ISO	2.89(0.35)	19.9	2923(603)	6.2	1.37(0.23)	18.2
	Gaussian DIAG	0.83(0.20)	65.5	3243(763)	14.2	0.86(0.17)	27.5
	Polynomial	2.50(0.28)	36.5	1109(362)	6.9	0.89(0.18)	28.0
	Perceptron	2.90(0.49)	18.5	5637(902)	4.8	0.64(0.19)	6.4
	G-DIAG+pol	0.89(0.23)	123.6	1886(489)	20.7	0.77(0.15)	36.1
	G-DIAG+pol+per	1.12(0.34)	143.6	2773(605)	23.4	0.60(0.12)	38.2
Roth ²	SVM	2.92		4140		2.02	
	RVM	2.80		3505		1.64	
	LASSO	2.84		3808		1.92	
MARS ³	mi=1	0.61(0.26)		14147(1312)		5.10(3.60)	
	mi=2	0.77(0.49)		2479(1021)		0.61(0.14)	
	mi=10,4,4	0.86(0.40)		2625(1312)		0.63(0.25)	

Table 3.5: mean minimum MSE over 100 trials, and standard deviations, on the three Friedman functions, using ASD with different features dictionaries.

¹ on one experiment

² from (Roth, 2004)

³ from (Friedman, 1991)

Conclusion

Throughout this thesis, we have re-formulated existing algorithms for solving the LASSO, following a slightly extended definition of this problem. The re-formulation of the homotopy algorithm, which was acknowledged as the standard LASSO-solving method, and was responsible for the popularity of this regression method and ℓ^1 regularization in general, has simplified further its description and actual implementation; this allowed the homotopy to exhibit running times that are equivalent or better than those of the coordinate descent method, which was recently believed to run significantly faster. Moreover, it has facilitated the analysis of degenerate cases where cycles or wrong feature selection may occur, and their workarounds. These workarounds have been based on deterministic methods rather than on stochastic perturbations, as was suggested prior to our work, and ultimately consist in very simple tests. Their derivations was made possible by exhibiting the familiarity of the homotopy and active set descent methods, and the possibility of combining them. The active set descent algorithm itself was given a simple and clear formulation and naming, that should help its understanding and recognition in the Machine Learning community. One of the direct benefits of this better understanding is its straightforward application to the regularized formulation of the LASSO, which is the most practical one. Some advantages over the homotopy method have been shown: a slightly lower computational complexity; more flexibility by not having to follow the regularization path, which prevents it from degenerative situations and permits to track non-stationary problems; ability to be applied to loss functions of which the regularization path is not linear. The flexibility, associated to the more regular form of the function to be optimized over the features, makes it a natural choice when applying our idea of performing the LASSO over a genuinely infinite feature set: the successive approximate optimizations that are then required are simpler, and imperfect optimizations do not have ill consequences. This simple idea of describing the observations by a whole multi-dimensional space of feature functions and letting the sole ℓ^1 regularization restrict this redundant description, and its very satisfying results, illustrates the practical, fresh and somehow candid approach with which we have tried to consider the problematics of regression.

More precisely, a deliberate choice not to build on the existing and much analyzed and well-grounded reproducing- kernel theories and applications, is at the origin of this approach. Most certainly, analyses and dissertations on the possible limitations of the kernel approach should be done. This would allow to assess the intuition that its combination of expressive power and computational reduction is in some sense deceiving. Indeed, a renouncement to exact computation while opting for a simple and integrated solution could be more fruitful than relying on the well-analyzable kernel methods with additional layers (kernel selection or multiple kernels). Theoretical analyzes of such refined extensions of kernel meth-

ods, and of the loss induced by approximation in the ECON approach, would help validate or moderate this intuition.

Other continuations of the present work include a more complete and detailed application of the active set descent method to alternative ℓ^1 -regularized loss functions, of which we have considered a few in chapter three. A full review and synthesis of the regression settings to which the algorithm applies would be a valuable contribution. Generalized linear models have been mentioned only through logistic regression, and a broader review would be needed as well.

The important question of model selection is reduced in the case of the LASSO to the right choice of the regularization parameter λ , and different ways of deciding this choice should be studied. In the general case, the parallel evolutions of the LASSO and restricted least-squares solutions seem to potentially give good indications of a switch from model-fitting to over-fitting. We have witnessed in some experiments, that the LASSO residual evolves regularly whereas there is a more noticeable change of behavior for the restricted least-squares when over-fitting occurs. Both theoretical and extended practical analyses are needed, to possibly identify a criterion that may be more robust than C_p -Mallow or cross-validation methods.

In the ECON setting, the differentiated penalizations of Gaussian features has given promising results on simple regression tasks, where the two fitting phases become distinguishably separated; theoretical and practical analyses should be conducted to confirm this profitable property.

Finally, applications of the active set descent to sequential learning problems certainly deserve further attention, especially for financial data flows and – according to my personal inclinations, in the domain of Reinforcement Learning.

Bibliography

The format of bibliographical entries is:

authors (year). title/chapter. *journal/book*, volume(issue):pages.

followed by the pages in which they are referenced in this thesis.

Alpaydin, E. (2005). *Introduction to Machine Learning*. MIT Press. 25

Armstrong, R. and Frome, E. (1976). A comparison of two algorithms for absolute deviation curve fitting. *Journal of the American Statistical Association*, 71(354):328–330. 154

Bach, F. (2008). Exploring large feature spaces with hierarchical multiple kernel learning. In *Neural Information Processing Systems*. 125

Bach, F., Thibaux, R., and Jordan, M. (2004). Computing regularization paths for learning multiple kernels. *Advances in Neural Information Processing Systems*, 17:41–48. 125

Bakin, S. (1999). *Adaptive regression and model selection in data mining problems*. Ph.d. thesis, The Australian National University. 151

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press. 147

Cornuéjols, A. and Miclet, L. (2010). *Apprentissage Artificiel: Concepts et algorithmes*. Eyrolles, 2nd edition. 25

Cover, T. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE transactions on electronic computers*, 14(3):326–334. 40

Dantzig, G. (1998). *Linear Programming and Extensions*. Princeton University Press. 63

Dantzig, G., Orden, A., and Wolfe, P. (1954). The generalized simplex method for minimizing a linear form under linear inequality restraints. *Pacific Journal of Mathematics*, 5:183–195. 58

Donoho, D. and Stodden, V. (2006). Breakdown Point of Model Selection When the Number of Variables Exceeds the Number of Observations. *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 1916–1921. 50

Draper, N. R. and Smith, H. (1966). *Applied Regression Analysis*. Wiley-Interscience. 46

- Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, 32(2):407–499. 26, 50, 76, 81, 110, 112, 153
- Friedman, J., Hastie, T., Höfling, H., and Tibshirani, R. (2007). Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332. 51, 90, 95, 96
- Friedman, J. H. (1991). Multivariate Adaptive Regression Splines. *The Annals of Statistics*, 19(1):1–67. 163
- Friedman, J. H., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22. 26, 120, 156
- Fu, W. J. (1998). Penalized Regressions: The Bridge versus the Lasso. *Journal of Computational and Graphical Statistics*, 7(3):397. 26, 51, 90
- Galton, F. (1886). Regression Towards Mediocrity in Hereditary Stature. *The Journal of the Anthropological Institute of Great Britain and Ireland*, 15:246–263. 31, 32, 34
- Gill, P. E., Golub, G. H., Murray, W. A., and Saunders, M. A. (1974). Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535. 87
- Golub, G. H. and Saunders, M. A. (1969). Linear least squares and quadratic programming. 87
- Guigue, V., Rakotomamonjy, A., and Canu, S. (2005). Kernel Basis Pursuit. *European Conference on Machine Learning 2005*, 20(6):757–774. 157
- Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J. (2008). *The elements of statistical learning: data mining, inference and prediction*. Springer, 2nd edition. 25, 144, 156
- Hayes, M. H. (1996). *Statistical digital signal processing and modeling*. John Wiley & Sons, Inc., 1st edition. 147
- Hoerl, A. E. and Kennard, R. W. (1970a). Ridge Regression: Applications to Nonorthogonal Problems. *Technometrics*, 12(1):69–82. 44
- Hoerl, A. E. and Kennard, R. W. (1970b). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1):55–67. 150
- Huber, P. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101. 155

- Jones, D. R., Perttunen, C. D., and Stuckman, B. E. (1993). Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181. 137
- Kim, S.-j., Lustig, M., Boyd, S., and Gorinevsky, D. (2007). An Interior-Point Method for Large-Scale ℓ_1 -Regularized Least Squares. *Journal on Selected Topics in Signal Processing*, 1(4):606–617. 26
- Kuhn, H. W. and Tucker, A. W. (1951). Nonlinear programming. In Neyman, J., editor, *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492. University of California Press, Berkeley, California. 68
- Lanckriet, G., Cristianini, N., Bartlett, P., Ghaoui, L., and Jordan, M. (2004). Learning the kernel matrix with semidefinite programming. *The Journal of Machine Learning Research*, 5:27–72. 125
- Lawson, C. and Hanson, R. (1974). *Solving Least Squares Problems*. John Wiley And Sons, Inc., New York. 50, 66, 144
- Lokhorst, J. (1999). The lasso and generalised linear models. Technical report, University of Adelaide. 156
- Loth, M., Davy, M., and Preux, P. (2007). Sparse temporal difference learning using lasso. In *Approximate Dynamic Programming and Reinforcement Learning, 2007. ADPRL 2007. IEEE International Symposium on*, pages 352–359. 148
- Loth, M. and Preux, P. (2007). A Unified View of TD Algorithms. In *European Symposium on Artificial Neural Networks (ESANN)*, pages 25–27. 148
- Loth, M. and Preux, P. (2010). The Iso-regularization Descent Algorithm for the LASSO. In Wong, K., Mendis, B., and Bouzerdoum, A., editors, *Neural Information Processing. Theory and Algorithms*, volume 6443 of *Lecture Notes in Computer Science*, pages 454–461. Springer Berlin / Heidelberg. 76
- Loth, M., Preux, P., Delepoulle, S., and Renaud, C. (2009). ECON: A Kernel Basis Pursuit Algorithm with Automatic Feature Parameter Tuning, and its Application to Photometric Solids Approximation. *Machine Learning and Applications, Fourth International Conference on*, 0:162–169. 124, 132, 141, 156, 159, 161
- Luo, Z. Q. and Tseng, P. (1992). On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35. 149
- Mallows, C. L. (1973). Some Comments on Cp. *Technometrics*, 15(4):661–675. 46

- Mika, S., Ratsch, G., Weston, J., Scholkopf, B., and Mullers, K. R. (1999). Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX, 1999.*, pages 41–48. IEEE. 157, 158
- Mitchell, T. M. (2006). *The Discipline of Machine Learning*. Technical report, Carnegie Mellon University, Pittsburgh, PA, USA. 25
- Nash, S. G. and Sofer, A. (1996). *Linear and nonlinear programming*. McGraw-Hill (New York). 68
- Nocedal, J. and Wright, S. J. (2000). *Numerical Optimization*. Springer. 59, 68
- Osborne, M., Presnell, B., and Turlach, B. (2000a). A new approach to variable selection in least squares problems. *IMA Journal of Numerical Analysis*, 20(3):389. 26, 50, 55, 66, 71, 73, 76, 78, 83, 85, 110, 112, 114
- Osborne, M., Presnell, B., and Turlach, B. A. (2000b). On the LASSO and Its Dual. *Journal of Computational and Graphical Statistics*, 9(2):319 – 337. 55, 71, 75, 76, 105, 156
- Qiu, S. and Lane, T. (2005). Multiple kernel learning for support vector regression. *Computer Science Department, The University of New Mexico, Albuquerque, NM, USA, Tech. Rep.*, pages 1–17. 125
- Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*. MIT Press. 41, 125
- Rätsch, G. (web). Benchmark repository. <http://www.fml.tuebingen.mpg.de/Members/raetsch/benchmark>. 157, 158
- Rätsch, G., Onoda, T., and Müller, K. (2001). Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320. 156, 157, 158
- Rosset, S., Swirszcz, G., Srebro, N., and Zhu, J. (2007). ℓ_1 regularization in infinite dimensional feature spaces. In Bshouty, N. and Gentile, C., editors, *Learning Theory*, volume 4539 of *Lecture Notes in Computer Science*, pages 544–558. Springer Berlin / Heidelberg. 119, 122
- Rosset, S. and Zhu, J. (2007). Piecewise linear regularized solution paths. *Annals of Statistics*, 35(3):1012. 149, 155, 156
- Roth, V. (2004). The generalized LASSO. *IEEE transactions on neural networks*, 15(1):16–28. 157, 158, 163
- Scholkopf, B. and Smola, A. J. (2001). *Learning With Kernels*. MIT Press Cambridge, MA, USA. 41
- Seeger, M. (2008). Low rank updates for the Cholesky decomposition. Technical report, University of California at Berkeley. 87

- Smola, A. J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222. 125
- Szepesvári, C. (2010). *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers. 147
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288. 31, 47, 50, 66, 76
- Tibshirani, R. (1997). The lasso method for variable selection in the Cox model. *Statistics in medicine*, 16(4):385–95. 26
- Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector machine. *J. Mach. Learn. Res.*, 1:211–244. 156
- Wang, L., Gordon, M., and Zhu, J. (2006). Regularized least absolute deviations regression and an efficient algorithm for parameter tuning. In *Data Mining, 2006. ICDM '06. Sixth International Conference on*, pages 690–700. 155
- White, H. (1980). A heteroskedasticity-consistent covariance matrix estimator and a direct test for heteroskedasticity. *Econometrica: Journal of the Econometric Society*, 48(4):817–838. 147
- Wolfe, P. (1963). A Technique for Resolving Degeneracy in Linear Programming. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):205–211. 112
- Xu, Z., Zhang, H., Wang, Y., Chang, X., and Liang, Y. (2010). L 1/2 regularization. *Science China Information Sciences*, 53(6):1159–1169. 50
- Yuan, M. and Lin, Y. (2006). Model Selection and Estimation in Regression with Grouped Variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67. 151, 152, 153
- Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320. 150

