Approche de Douglas-Rachford aléatoire par blocs appliquée à la régression logistique parcimonieuse

Giovanni Chierchia[1], Afef Cherni[1,2], Emilie Chouzenoux[1,3], Jean-Christophe Pesquet[3]

[1]Université Paris Est, LIGM UMR 8049, CNRS, ENPC, ESIEE Paris, UPEM, Noisy-le-Grand, France.

[2]IGBMC, CNRS UMR 7104, Inserm U 964, Illkirch-Graffenstaden, France.

[3]Centre pour la Vision Numérique, CentraleSupélec, Université Paris-Saclay, Châtenay-Malabry, France.

Résumé − Dans cet article, nous proposons un nouvel algorithme d'optimisation stochastique pour la régression logistique parcimonieuse, basé sur une version aléatoire par blocs de l'algorithme de Douglas-Rachford. Notre algorithme parcourt la base d'apprentissage en sélectionnant aléatoirement un bloc de données à chaque itération, puis réalise l'étape de mise à jour en utilisant l'opérateur proximal de la fonction de coût logistique, dont nous donnons une expression explicite. Les tests réalisés sur plusieurs jeux de données démontrent l'efficacité de notre algorithme par rapport aux méthodes de type gradient stochastique.

Abstract − We propose a stochastic optimization algorithm for logistic regression based on a randomized version of Douglas-Rachford splitting method. Our algorithm sweeps the training set by randomly selecting a mini-batch of data at each iteration, and it performs the update step by leveraging the proximity operator of the logistic loss, for which a closed-form expression is derived. Experiments carried out on standard datasets compare the efficiency of our algorithm to stochastic gradient-like methods.

1 Introduction

Les méthodes d'apprentissage parcimonieux sont devenues très populaires dans le contexte de l'apprentissage supervisé, de par leur capacité à éliminer les caractéristiques non pertinentes pendant la phase d'entrainement. Ces approches visent à apprendre une combinaison linéaire pondérée de fonctions de base liant les données d'entraînement, en forçant autant de poids que possible à être nuls. Ceci revient à résoudre un problème d'optimisation qui met en jeu une fonction de coût et une régularisation parcimonieuse. Différents classifieurs sont obtenus en faisant varier la fonction de coût, les plus populaires étant la fonction *hinge* et la fonction logistique [20].

Dans le cadre de l'apprentissage supervisé, l'utilisation de régularisations parcimonieuses remonte aux travaux de Bradley et Mangasarian [3], qui ont montré qu'une sélection efficace des fonctions de bases était obtenue grâce à un seuillage à zéro des faibles poids, ce qui revient à l'utilisation d'une régularisation ℓ_1. D'autres formes de régularisations ont été également étudiées [10, 14].

De nombreux algorithmes d'apprentissage existent pour traiter le cas de la régularisation quadratique, exploitant la forme simple du dual du problème au sens de Lagrange [2]. Cependant, une telle approche perd son avantage en présence d'une régularisation parcimonieuse, car le problème dual devient dans ce cas aussi difficile à traiter que le primal. Les classifieurs linéaires parcimonieux sont donc habituellement entraînés en résolvant directement le problème d'optimisation primal associé. Parmi les algorithmes possibles, nous pouvons mentionner les méthodes de majoration-minimisation [12] et les approches proximales [1, 5, 10].

En pratique, l'usage de règles de mise à jour stochastiques permet de réduire significativement le temps de calcul des méthodes d'apprentissage supervisé, lorsqu'une régularisation quadratique est employée [2]. De nombreux travaux récents ont visé à généraliser ces méthodes d'optimisation stochastiques pour pouvoir traiter des régularisations favorisant la parcimonie [18]. Citons, en particulier, les stratégies de minimisation par blocs [22], les approches explicites-implicites stochastiques [8, 21], les algorithmes primaux-duaux stochastiques [9, 19], les techniques de descente duale stochastiques [24], et les méthodes de majoration-minimisation stochastiques [6, 13].

Parmi toutes ces approches, les algorithmes primaux-duaux stochastiques sont en fait les seules techniques permettant de s'affranchir de l'hypothèse de différentiabilité de la fonction de coût. Ils permettent en effet de traiter toute fonction de coût convexe par son opérateur proximal, en supposant que celui-ci puisse être calculé efficacement. C'est le cas de la fonction *hinge*, mais pas de la régression logistique, pour laquelle il n'existe pas d'expression explicite dans la littérature.

Dans cet article, nous proposons un algorithme proximal stochastique pour la régression logistique parcimonieuse,

basé sur un schéma de Douglas-Rachford. Notre approche traite la fonction de coût logistique par son opérateur proximal, ce qui contraste avec les méthodes existantes utilisant plutôt son gradient. De ce fait, l'algorithme proposé n'est pas contraint par la valeur de la constante de Lipschitz du gradient de la fonction de coût, ce qui, par conséquent, peut conduire à une convergence plus rapide. D'un point de vue théorique, notre contribution principale est de montrer que l'opérateur proximal de la fonction logistique peut en fait s'exprimer sous une forme quasi explicite, et ainsi être calculé efficacement. Des comparaisons avec plusieurs approches d'optimisation stochastique de l'état de l'art sont réalisées sur des bases de données de référence.

L'article s'organise comme suit. Dans la section 2, nous explicitons la formulation du problème de régression logistique parcimonieuse. Dans la section 3, nous décrivons l'algorithme proposé, et dérivons l'expression de l'opérateur proximal de la fonction logistique. Dans la section 4, nous évaluons les performances de notre méthode, et la comparons avec trois algorithmes de classification parcimonieuse de la littérature [4, 21, 24]. Enfin, la section 5 conclut ce travail et dresse quelques perspectives.

Notations: $\Gamma_0(\mathcal{H})$ représente l'ensemble des fonctions propres, convexes, semi-continues inférieurement d'un espace de Hilbert $\mathcal{H}$ vers $\mathbb{R} \cup \{+\infty\}$. Pour tout $\nu \in \mathcal{H}$, le sous-différentiel de $\psi \in \Gamma_0(\mathcal{H})$ en ν est noté $\partial\psi(\nu) = \{\xi \in \mathcal{H} \mid (\forall \zeta \in \mathcal{H})\ \langle \zeta - \nu \mid \xi \rangle + \psi(\nu) \leq \psi(\zeta)\}$.

2 Régression logistique

Un classifieur linéaire binaire vise à associer, à chaque donnée en entrée $x \in \mathbb{R}^N$, une valeur binaire $y \in \{-1, +1\}$ via un modèle linéaire de la forme:

$$d_w(x) = \text{signe}\left(x^\top w\right). \tag{1}$$

En apprentissage supervisé, le vecteur de poids $w \in \mathbb{R}^N$ défini ci-dessus est estimé à partir d'un ensemble de paires entrée/sortie:

$$\mathcal{S} = \left\{(x_\ell, y_\ell) \in \mathbb{R}^N \times \{-1, +1\} \mid \ell \in \{1, \dots, L\}\right\}, \tag{2}$$

qu'on appelle *la base d'entrainement*.

Une approche très populaire en apprentissage pour effectuer la tâche de prédiction se base sur la notion de risque structurel. L'objectif est de réaliser un compromis entre la fidélité à la base d'entrainement et la réduction de la complexité du modèle, en résolvant le problème d'optimisation suivant:

$$\underset{w \in \mathbb{R}^N}{\text{minimiser}}\ f(w) + \sum_{\ell=1}^{L} h\left(y_\ell\, x_\ell^\top w\right), \tag{3}$$

où $f \in \Gamma_0(\mathbb{R}^N)$ est une fonction de régularisation, tandis que $h \in \Gamma_0(\mathbb{R})$ est la fonction de coût.

De nombreux choix ont été proposés dans la littérature pour la fonction de régularisation. Dans le contexte de l'apprentissage parcimonieux, un choix courant est d'utiliser la norme ℓ_1, $f = \lambda\|\cdot\|_1$, avec $\lambda > 0$. Pour la fonction de coût, là encore, plusieurs choix sont possibles, tels que, dans le cas de la classification, la fonction *hinge*, la fonction *hinge* carrée, ou le coût logistique, et, dans le cas de la régression, le coût quadratique ou le coût de Huber [20]. Dans ce travail, nous nous intéressons plus particulièrement au coût logistique, défini par :

$$(\forall v \in \mathbb{R}) \qquad h(v) = \log\Big(1 + \exp(-v)\Big). \tag{4}$$

3 Méthode d'optimisation

La résolution du problème (3) requiert un algorithme d'optimisation capable de traiter un terme de régularisation non différentiable favorisant la parcimonie. Dans ce contexte, les méthodes proximales apparaissent comme une solution de choix, de par leur capacité à gérer efficacement des problèmes convexes non lisses [7, 17]. Ces méthodes reposent sur un outil fondamental, appelé *l'opérateur proximal* [16], défini pour une fonction $\psi \in \Gamma_0(\mathcal{H})$ comme

$$(\forall \zeta \in \mathcal{H}) \quad \text{prox}_\psi(\zeta) = \underset{\xi \in \mathcal{H}}{\text{argmin}}\ \frac{1}{2}\|\xi - \zeta\|^2 + \psi(\xi). \tag{5}$$

L'opérateur proximal peut être interprété comme une étape de sous-gradient implicite, sur la fonction ψ. En effet, $\nu = \text{prox}_\psi(\zeta)$ est défini de façon unique par l'inclusion $\zeta - \nu \in \partial\psi(\nu)$. De nombreux algorithmes proximaux existent dans la littérature, permettant de résoudre différentes classes de problèmes. Dans ce travail, nous proposons d'utiliser une version aléatoire par blocs de l'algorithme proximal de Douglas-Rachford, introduite dans [8]. Un aspect remarquable de cet algorithme est qu'il permet une mise à jour par blocs des variables du problème. Ici, nous nous intéressons à une mise à jour partielle de variables auxiliaires qui nous permet de travailler à chaque itération sur un sous-ensemble des données de la base d'entrainement, plutôt que sur la base complète. Une telle stratégie se rapproche en ce sens des méthodes de gradient incrémental / stochastique reconnues pour leur efficacité dans les problèmes d'apprentissage supervisé.

Nous obtenons, dans le cadre de la résolution du Problème (3), l'algorithme 1. A chaque itération i de la méthode, un sous-ensemble $\mathbb{L}_i$ de $\{1, \dots, L\}$ est sélectionné, de façon aléatoire, et seules les composantes $(v_\ell^{[i]})_{\ell \in \mathbb{L}_i}$ de la variable auxiliaire v sont mises à jour, à partir des données $(x_\ell, y_\ell)_{\ell \in \mathbb{L}_i}$ extraites de la base. Notons que le calcul de la matrice Q définie dans l'algorithme 1 est généralement peu coûteux dans les applications en apprentissage, la dimension N étant souvent petite devant L. Les approches proximales présentent l'avantage de posséder des propriétés de convergence solides, même dans un contexte stochastique. Dans notre cas, nous pouvons déduire du résultat de [8] que, si la probabilité d'activation de chaque variable est non nulle et indépendante de i, sous des hypothèses

de qualification du problème, la suite $(w^{[i]})_{i\in\mathbb{N}}$ générée par l'algorithme 1 converge presque sûrement vers une solution du Problème (3).

Algorithm 1 Algorithme de Douglas-Rachford aléatoire par blocs

INITIALISATION

$$\left|\begin{array}{l}\text{Calculer } \; Q=\Big(\mathrm{Id}+\sum_{\ell=1}^{L} x_\ell x_\ell^\top\Big)^{-1}\\ \text{Choisir } \; t^{[0]}\in\mathbb{R}^N \;\text{ et }\; \big(v_1^{[0]},\dots,v_L^{[0]}\big)\in\mathbb{R}^L\\ \text{Fixer } \; u^{[0]}=\sum_{\ell=1}^{L} y_\ell x_\ell\, v_\ell^{[0]}.\\ \text{Choisir } \; \gamma\in\,]0,+\infty[\;\text{ et }\; \mu\in\,]0,2[\end{array}\right.$$

POUR $i=0,1,\dots$

$$\left|\begin{array}{l}\text{Sélectionner } \; \mathbb{L}_i\subset\{1,\dots,L\}\\ w^{[i]}=Q\big(t^{[i]}+u^{[i]}\big)\\ t^{[i+1]}=t^{[i]}+\mu\Big(\mathrm{prox}_{\gamma_i f}(2w^{[i]}-t^{[i]})-w^{[i]}\Big)\\ (\forall \ell\in\mathbb{L}_i)\quad v_\ell^{[i+1]}=v_\ell^{[i]}\\ \qquad\qquad +\mu\Big(\mathrm{prox}_{\gamma_i h}(2y_\ell x_\ell^\top w^{[i]}-v_\ell^{[i]})-y_\ell x_\ell^\top w^{[i]}\Big)\\ (\forall \ell\notin\mathbb{L}_i)\quad v_\ell^{[i+1]}=v_\ell^{[i]}\\ u^{[i+1]}=u^{[i]}+\sum_{\ell\in\mathbb{L}_i}\big(v_\ell^{[i+1]}-v_\ell^{[i]}\big)y_\ell x_\ell.\end{array}\right.$$

A chaque itération de l'algorithme 1, il est nécessaire d'évaluer les opérateurs proximaux de f et h. De ce fait, un calcul rapide de ces opérateurs est nécessaire pour garantir de bonnes performances numériques de l'algorithme. Un grand nombre de fonctions présentent une forme explicite pour leur opérateur proximal [7, 17]. Par exemple, si f est la norme ℓ_1, son opérateur proximal se réduit à un simple seuillage doux. En ce qui concerne la fonction de coût logistique, quelques stratégies de calcul numérique ont été proposées [17, 23], mais à notre connaissance, une forme explicite de son opérateur proximal est inconnue. La contribution principale de cet article, présentée dans la proposition ci-dessous, permet de combler cette lacune.

Proposition 3.1. *Soit $\gamma\in\,]0,+\infty[$. L'opérateur proximal de la fonction de coût logistique* (4) *est*

$$(\forall v\in\mathbb{R})\quad \mathrm{prox}_{\gamma h}(v)=v+\mathrm{W}_{\exp(-v)}\Big(\gamma\exp\left(-v\right)\Big), \quad (6)$$

où W. *est la fonction de* W*-Lambert généralisée [11, 15].*

La fonction de W-Lambert généralisée (aussi appelée fonction de r-Lambert) mentionnée ci-dessus permet de résoudre les équations transcendantales de la forme $p\exp(p)+rp=q$, où $p=W_r(q)$. Elle peut être évaluée de façon rapide et très précise, par la méthode de type Newton développée par Mező *et al.* [15], disponible en ligne[1].

4 Résultats expérimentaux

Nous évaluons maintenant les performances de notre approche sur les bases de données MNIST[2] ($N=717$ et $L=60000$) et W8A[3] ($N=300$ et $L=49749$). Nous comparons notre algorithme, avec les méthodes suivantes : l'algorithme stochastique explicite-implicite (SFB), l'approche duale régularisée moyennée (RDA) [24], et l'algorithme primal-dual par blocs (BCPD) [4]. Pour chacun de ces algorithmes, nous traçons l'évolution du critère défini en (3) en fonction du temps de calcul.

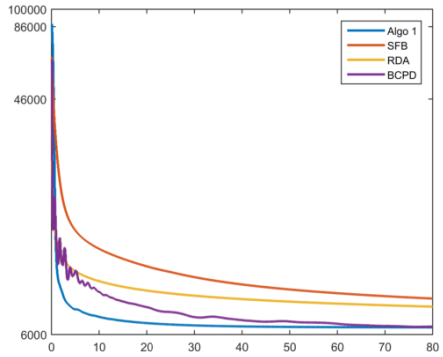
Les paramètres des différentes méthodes sont fixés manuellement afin d'optimiser leurs performances : le *taux d'apprentissage* de SFB et RDA est $\gamma=10^{-4}$, tandis que $\gamma=1$ et $\mu=1.8$ pour l'algorithme 1. Des blocs de données de taille 1000 sont sélectionnés aléatoirement à chaque itération, suivant une loi uniforme, et une régularisation ℓ_1 pondérée par $\lambda=1$ est employée. Tous les tests ont été menés sous Matlab 2015a, avec un CPU Intel i7 cadencé à 3.40 GHz et doté de 12 Go de RAM.

La Figure 1 illustre les performances des différentes méthodes, pour les deux jeux de données testés. Les résultats indiquent que notre méthode permet d'atteindre une convergence rapide vers une valeur faible de la fonction objectif. Ceci s'explique en partie par le rôle de préconditionneur joué par la matrice Q intégrée dans l'algorithme 1. Un point notable est que notre méthode laisse un choix libre pour les paramètres γ et μ. Cela contraste avec les méthodes SFB et RDA, où le paramètre γ doit être attentivement réglé à la main, sans quoi les méthodes deviennent très lentes, ou bien à l'inverse, divergent.

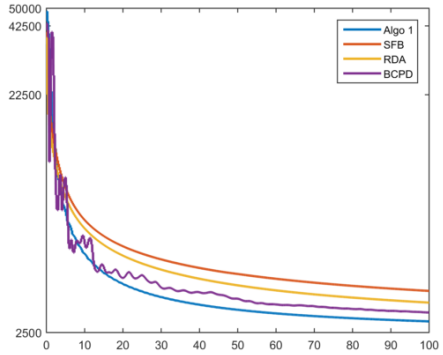
5 Conclusion

Dans cet article, nous avons proposé un algorithme de Douglas-Rachford aléatoire par blocs, pour résoudre un problème de régression logistique parcimonieuse. Notre approche se base sur l'opérateur proximal de la fonction de coût logistique, pour lequel nous avons obtenu une expression explicite facile à évaluer. Ces contributions nous ont permis de mettre en place une méthode de résolution rapide, et dont les paramètres de pas peuvent être choisis de manière flexible, sans risquer de faire diverger l'algorithme. Ceci constitute un avantage par rapport aux techniques d'optimisation stochastique usuelles en apprentissage. Nos résultats expérimentaux confirment les bonnes performances de notre algorithme.

[1] https://sites.google.com/site/istvanmezo81/others
[2] yann.lecun.com/exdb/mnist
[3] www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html



(a) Dataset W8A : $N = 300$ et $L = 49749$.



(b) Dataset MNIST : $N = 717$ et $L = 60000$.

Figure 1: Critère (3) en fonction du temps.

Remerciements

Ce travail a été financé par le défi CNRS MASTODONS 2016TABASCO.

Références

[1] M. Barlaud, W. Belhajali, P. L. Combettes, and L. Fillatre. Classification and regression using a constrained convex splitting method. *IEEE Trans. Signal Process.*, 2017.

[2] M. Blondel, A. Fujino, and N. Ueda. Large-scale multiclass support vector machine training via euclidean projection onto the simplex. In *Proc. of ICPR*, pages 1289–1294, Stockholm, Sweden, 24-28 August 2014.

[3] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In *Proc. of ICML*, pages 82–90, Madison, USA, 1998.

[4] G. Chierchia, N. Pustelnik, and J.-C. Pesquet. Random primal-dual proximal iterations for sparse multiclass SVM. In *Proc. of MLSP*, Salerno, Italy, September 2016.

[5] G. Chierchia, N. Pustelnik, J.-C. Pesquet, and B. Pesquet-Popescu. A proximal approach for sparse multiclass SVM. *Preprint arXiv:1501.03669*, February 2015.

[6] E. Chouzenoux and J.-C. Pesquet. A stochastic majorize-minimize subspace algorithm for online penalized least squares estimation. *IEEE Trans. Signal Process.*, 2017.

[7] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer-Verlag, New York, 2011.

[8] P. L. Combettes and J.-C. Pesquet. Stochastic quasi-Fejér block-coordinate fixed point iterations with random sweeping. *SIAM J. Opt.*, 25(2):1221–1248, July 2015.

[9] P.L. Combettes and J.-C. Pesquet. Stochastic approximations and perturbations in forward-backward splitting for monotone operators. *Pure and Applied Functional Analysis*, 1(1):13–37, January 2016.

[10] L. Laporte, R. Flamary, S. Canu, S. Déjean, and J. Mothe. Non-convex regularizations for feature selection in ranking with sparse SVM. *IEEE Trans. Neural Netw. Learn. Syst.*, 25(6):1118 – 1130, June 2014.

[11] A. Maignan and T. Scott. Fleshing out the generalized lambert W function. *ACM Communications in Computer Algebra*, 50(2):45–60, June 2016.

[12] J. Mairal. Optimization with first-order surrogate functions. In *Proc. of ICML*, pages 783–791, 2013.

[13] J. Mairal. Stochastic majorization-minimization algorithms for large-scale optimization. In *Proc. of NIPS*, pages 2283–2291, 2013.

[14] L. Meier, S. Van De Geer, and P. Bühlmann. The group Lasso for logistic regression. *J. R. Stat. Soc. B*, 70(1):53–71, 2008.

[15] I. Mező and Á. Baricz. On the generalization of the lambert W function. *Trans. Amer. Math. Soc.*, 2017.

[16] J. J. Moreau. Proximité et dualité dans un espace hilbertien. *Bull. Soc. Math. France*, 93:273–299, 1965.

[17] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2014.

[18] M. Pereyra, P. Schniter, E. Chouzenoux, J.-C Pesquet, J.-Y. Tourneret, A. Hero, and S. McLaughlin. A survey of stochastic simulation and optimization methods in signal processing. *IEEE J. Sel. Topics Signal Process.*, 10(2):224–241, March 2016.

[19] J.-C. Pesquet and A. Repetti. A class of randomized primal-dual algorithms for distributed optimization. *J. Nonlinear Convex Anal.*, 16(12):2453–2490, 2015.

[20] L. Rosasco, E. De Vito, A. Caponnetto, M. Piana, and A. Verri. Are loss functions all the same? *Neural Comput.*, 16(5):1063–1076, May 2004.

[21] L. Rosasco, S. Villa, and B. C. Vũ. Stochastic forward-backward splitting for monotone inclusions. *J. Optim. Theory Appl.*, 169(2):388–406, May 2016.

[22] S. Shalev-Shwartz and A. Tewari. Stochastic methods for l1-regularized loss minimization. *J. Mach. Learn. Res.*, 12:1865–1892, June 2011.

[23] P.-W. Wang, M. Wytock, and J. Z. Kolter. Epigraph projections for fast general convex programming. In *Proc. of ICML*, 2016.

[24] L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *J. Mach. Learn. Res.*, 11:2543–2596, October 2010.