Active set strategy for high-dimensional non-convex sparse optimization problems

Aurélie Boisbunon, Rémi Flamary, Alain Rakotomamonjy

ACTIVE SET STRATEGY FOR HIGH-DIMENSIONAL NON-CONVEX SPARSE OPTIMIZATION PROBLEMS

Aurélie Boisbunon[⋆] *Rémi Flamary*[⋆⋆] *Alain Rakotomamonjy*[†]

[⋆] INRIA Sophia-Antipolis Méditerranée – AYIN Team
aurelie.boisbunon@inria.fr
[⋆⋆] Université de Nice Sophia-Antipolis – Lagrange Laboratory
remi.flamary@unice.fr
[†] Université de Rouen – LITIS Laboratory
alain.rakoto@univ-rouen.fr

ABSTRACT

The use of non-convex sparse regularization has attracted much interest when estimating a very sparse model on high dimensional data. In this work we express the optimality conditions of the optimization problem for a large class of non-convex regularizers. From those conditions, we derive an efficient active set strategy that avoids the computing of unnecessary gradients. Numerical experiments on both generated and real life datasets show a clear gain in computational cost *w.r.t.* the state of the art when using our method to obtain very sparse solutions.

***Index Terms*—** Non-convex optimization, sparsity, very large scale

1. INTRODUCTION

Sparsity and sparse optimization have been of particular interest to the machine learning and signal processing community in the last decades, especially with the exponential increase in the size of datasets. One of the most successful strategies for sparse estimation is the use of non-differentiable sparsity promoting regularizers in the optimization framework, more specifically the ℓ_1 regularization also known as Lasso [1, 2]. However, it is well known that the ℓ_1-penalty is biased and not always consistent [3]. The interest has thus been shifted toward other types of penalties, in particular non-convex and non-differentiable ones. Indeed, the main advantage of the latter is that they reduce the bias of the ℓ_1-penalty while possibly leading to even sparser solutions. Hence, non-convex penalties are especially useful in datasets where we wish to select only a small set of features out of a huge number. Popular examples are the Smoothly Clipped Absolute Deviation (SCAD) [4], the Minimax Concave Penalty (MCP) [5] and the Log-Sum Penalty (LSP) [6] (see [7] for more examples).

The non-convexity obviously makes the problem harder to solve and does not necessarily yield a unique solution. Nevertheless, there have been recent works proposing efficient algorithms: among others, the Difference of Convex (DC) programming [8] is actually equivalent to a reweighted ℓ_1 penalty [9, 6]; the General Iterative Shrinkage and Threshold (GIST) algorithm [7] is based on proximal operators for non-convex penalties and the Sequential Convex Programming (SCP) [10] uses local approximations at each iteration.

As mentioned before, non-convexity is especially interesting in very large datasets where we wish to obtain a very sparse solution. It was shown in [7] that the GIST algorithm converges faster and is more efficient than DC and SCP. Even so, GIST is a gradient descent method that typically requires a large number of gradient computations and can thus be costly when dealing with more than several hundreds of thousands of features. This is the case for instance for datasets in pharmakinetics, genetics, social networks or recommendation systems [11]. It is thus highly desirable to modify the existing algorithms so that they can handle that many features in a reasonable time.

We propose in this work to use an active set strategy for handling an extremely large number of features. The main idea behind our approach is that when the solution is very sparse, one should avoid dealing with all the variables at each iteration, as done in a descent algorithm. The gradient computation on all variables is still necessary but will be computed only periodically in order to add variables to the active set. The rest of this communication is organized as follows. We establish optimality conditions for a large family of non-convex regularizers in Section 2 and propose an active set algorithm that uses these conditions in Section 3. Finally, Section 4 displays the results of numerical experiments.

A. Boisbunon would like to thank the French Spatial Agency (CNES) for the financial support of her post-doc at INRIA.

R. Flamary acknowledges support of CNRS MASTODONS project DISPLAY Distributed processing for very large arrays in radioastronomy.

A. Rakotomamonjy acknowledges support of the French ANR under the grant 12-BS02-004

2. LEARNING PROBLEM

2.1. Framework

As stated in the introduction, we are interested in solving sparse optimization problems that occur for instance in machine learning or in compressed sensing problems. Such problems are often of the form

$$\min_{\mathbf{x}\in\mathbb{R}^p} \{f(\mathbf{x}) = l(\mathbf{x}) + r(\mathbf{x})\}, \tag{1}$$

where $l(\cdot)$ is a proper and differentiable function with a Lipschitz-continuous gradient and $r(\cdot)$ is a proper, lower semi-continuous and possibly non-convex function. The function l is usually taken as the data-fitting term, like the least-squares or the logistic loss functions. Here, the function r corresponds to the penalty or regularization term leading to a sparse solution, like the SCAD or the log-sum penalties.

We further assume that the sparsity-inducing regularization term can be expressed as the difference of two convex functions r_1 and r_2, that is, $r(\mathbf{x}) = r_1(\mathbf{x}) - r_2(\mathbf{x})$. In addition, we also assume that the functions r_1 and r_2 can be expressed as

$$r_1(\mathbf{x}) = \sum_i g_1(|x_i|), \quad r_2(\mathbf{x}) = \sum_i g_2(|x_i|), \tag{2}$$

with $g_1(\cdot)$ and $g_2(\cdot)$ two differentiable functions on $[0,\infty)$. Note that most regularization terms of interest can be rewritten under this form, hence this hypothesis barely induces loss of generality. For instance, taking $g_1(|x_i|) = \lambda|x_i|$ and $g_2(|x_i|) = \lambda|x_i| - \lambda\log(1 + |x_i|/\theta)$ yields the log-sum penalty, while $g_1(|x_i|) = \lambda|x_i|$ and $g_2(|x_i|) = \lambda|x_i| - \lambda(1 + |x_i|/\theta)^p$, $0 < p < 1$, gives the ℓ_p regularization term, where $\theta, \lambda > 0$ in both cases.

2.2. DC Optimality conditions

According to the above framework, we seek at the optimality condition for a difference of convex functions where the first convex part is $l(\mathbf{x}) + r_1(\mathbf{x})$ and the second one is $r_2(\mathbf{x})$, thus corresponding to a DC framework. DC programming has already been intensively analyzed and some chararacterizations of local minima have been provided. For instance, if $\mathbf{x}^*$ is a local minimum of problem (1), then we have [12, Thm 3.4.6]

$$\partial r_2(\mathbf{x}^*) \subset \nabla l(\mathbf{x}^*) + \partial r_1(\mathbf{x}^*), \tag{3}$$

where $\nabla(\cdot)$ denotes the gradient operator and $\partial(\cdot)$ the subdifferential operator. Note that, since r_1 and r_2 are convex, their subdifferentials at any point $\mathbf{x}$ are always non-empty [13]. The above conditions can be rewritten as follows: $\forall \mathbf{z}_2 \in \partial r_2(\mathbf{x}^*), \exists\, \mathbf{z}_1 \in \partial r_1(\mathbf{x}^*)$ such that

$$0 = \nabla l(\mathbf{x}^*) + \mathbf{z}_1 - \mathbf{z}_2.$$

From the separability of the regularization term $r(\cdot)$, the optimality condition can be expressed componentwise. Indeed, denoting the ith component of $\mathbf{a}$ (resp. $\mathbf{z}_k$) by a_i (resp. $z_{k,i}$), we obtain

$$0 = -\nabla l(\mathbf{x})_i + z_{1,i} - z_{2,i}. \tag{4}$$

Further, applying the subgradient chain rule yields $z_{k,i} = g_k'(|x_i|)\beta_k$, $k \in \{1,2\}$, with $\beta_k \in \partial|x_i|$. Since the subdifferential of $|x_i|$ is well known to be $[-1,1]$ for $x_i = 0$ and $s_i = sign(x_i)$ otherwise, there exists β_1 such that, $\forall\beta_2$, the final optimality conditions are

$$\nabla l(\mathbf{x})_i = \begin{cases} \big(g_1'(|x_i|) - g_2'(|x_i|)\big)s_i, & x_i \neq 0 \\ \beta_1 g_1'(0) - \beta_2 g_2'(0), & x_i = 0. \end{cases} \tag{5}$$

The optimality condition for $x_i = 0$ is the one that helps achieve sparsity. It can be further simplified depending on g_1 and g_2 after simple algebras. For instance, it can be shown that if $g_2'(0) = 0$ then this condition becomes

$$|\nabla l(\mathbf{x})_i| \leq g_1'(0) \quad \text{if} \quad x_i = 0. \tag{6}$$

For g_2 of the form $g_2 = g_1 - h$, then this condition becomes

$$|\nabla l(\mathbf{x})_i| \leq h'(0) \quad \text{if} \quad x_i = 0. \tag{7}$$

These conditions allow a better understanding of why the sparsity promoting terms discussed earlier are more agressive in terms of sparsity. Indeed, for a classical ℓ_1 regularization, we obtain Condition (6) with $g_1'(0) = \lambda$, which boils down to the classical ℓ_1 optimality conditions. But when using other regularization terms, g_1 or h tend to have larger derivatives in 0, making the condition for $x_i = 0$ easier to respect and thus inducing more sparsity.

For the sake of clarity, we explicitely derive the zero-component optimality condition for the log-sum penalty (LSP), where $r(\mathbf{x}) = \lambda\sum_{i=1}^p \log(1 + |x_i|/\theta)$. The difference of convex functions $r = r_1 - r_2$ can be simply taken as in Equation (2) with $g_1(y) = \lambda\, y$ and $g_2(y) = \lambda\{y - \log(1 + y/\theta)\}$. According to the above expression of g_1 and g_2 for LSP, g_2 is of the form $g_1 - h$, with $h(\cdot) = \log(1 + \cdot/\theta)$, thus Condition (7) becomes

$$|\nabla l(\mathbf{x})_i| \leq \lambda/\theta \quad \text{if} \quad x_i = 0. \tag{8}$$

For the ℓ_p penalty, for which g_2 has a similar form, the upper bound in Condition (7) is $\lambda p/\theta$. For the capped-ℓ_1 penalty, denoted as $r(\mathbf{x}) = \sum_{i=1}^p \lambda\min(|x_i|, \theta)$, we have the same expression for g_1, while $g_2(y) = \lambda\,(y - \theta)_+$. Since $g_2'(0) = 0$, the condition for a zero variable to be optimal is given by Equation (6) with $g_1'(0) = \lambda$.

Now that we have derived a necessary condition for a local minimum of Problem (1), we are able to propose our active set algorithm.

3. ALGORITHM

Our objective in this work is to derive an efficient algorithm that outputs a critical point satisfying the above-given conditions for very high-dimensional datasets. For this purpose, we

Algorithm 1 Active set algorithm for non-convex optimization based on optimality condition (7)

Inputs

- Initial active set $\varphi = \emptyset$

1: **repeat**
2: $\mathbf{x} \leftarrow$ Solve Problem (1) with current active set φ
3: Compute $\mathbf{r} \leftarrow |\nabla l(\mathbf{x})|$
4: **for** $k = 1, \ldots, k_s$ **do**
5: $j \leftarrow \arg\max_{i \in \bar{\varphi}} r_i$
6: If $r_j > h'(0) + \varepsilon$ then $\varphi \leftarrow j \cup \varphi$
7: **end for**
8: **until** stopping criterion is met

propose an active set strategy that alternatively solves a master problem, which correponds to Problem (1) restricted to a limited number of variables (the active set), and then adds to the active set inactive variables that violate the necessary condition given in Equation (6) or (7). The active set strategy is a classical tool in convex optimization [14] but its uses remain limited when it comes to non-convex optimization. As far as our knowledge goes, this is the first work that considers an active set approach for sparse efficient non-convex learning problems. This active set algorithm, given in Algorithm 1, can be easily inferred from Condition (6) or (7) and is very efficient when the solution of the optimization problem is extremely sparse. In a nutshell, we begin with the trivial solution $\mathbf{x}^{(0)} = \mathbf{0}$ and the corresponding active set $\varphi = \emptyset$ containing the set of variables such that $x_i \neq 0$. At each iteration, Problem (1) is solved on the data restricted to the active set φ using a solver such as GIST [7] or SCP [10]. Then the optimality conditions are checked and the k_s variables in the inactive set $\bar{\varphi}$ that violate the most those conditions are added to the active set. These steps are repeated until a convergence criterion is met. In this work we stopped the algorithm when all the optimality conditions on the unselected variables in $\bar{\varphi}$ are respected up to a tolerance ε.

In order to have a more efficient algorithm, several variables can be added to the active set at each iteration. The tolerance parameter ε also acts in practice as a thresholding parameter because variables that only slightly violate the optimality conditions tend to stay at zero. Finally note that an efficient warm-starting scheme can be used for solving the *master* problem (Line 2 of the algorithm). For instance, the GIST algorithm strongly benefits from initialization near optimality and can be used for solving this problem.

3.1. Related works and algorithms

Most approaches considered for solving the non-convex problem given in Equation (1) are based on a majorization-minimization (MM) approach. They usually consider a linear or quadratic majorization of the non-convex penalty [4, 15, 9]. Other works such as [16] can handle non-convexity in the data fitting term but keep a convex regularization. The recent GIST [7] or SCP approaches [10] slightly differ from the above cited works, as they also propose to majorize the loss function. They both consider a quadratic majorization of the differentiable loss function. The difference between GIST and SCP relies then on how the non-convex penalty is dealt with. SCP proposes again a majorization whereas GIST uses proximal operator for handling it. The common point of all these algorithms is that they have to deal with all the variables during the optimization. For very high-dimensional problems, this point can lead to computationally expensive algorithms.

On the contrary, the active set algorithm we propose optimizes Problem (1) only on a few number of variables (by using one of the above approaches, typically GIST) and afterwards adds some variables that may be non-zero at optimum. This strategy drastically reduces the computational complexity in high dimension.

4. NUMERICAL EXPERIMENTS

In order to illustrate the advantages of using an active set strategy for extremely sparse optimization problems, we tested our approach on a generated dataset with full matrices and two real life datasets with sparse feature matrices. In all experiments we considered the common quadratic data term $l(\mathbf{x}) = ||\mathbf{y} - \mathbf{A}\mathbf{x}||^2$ used for regression problems in the machine learning community and signal estimation in the signal processing community. The chosen non-convex regularization term is the log-sum penalty.

Note that in all the experiments the regularization parameter is set to $\theta = 0.1$ which leads to a strongly nonconvex function that promotes sparsity more aggressively than the ℓ_1 regularization with less bias. The stopping condition parameter ε is set to 0.1 and the maximum number of features k_s added to the active set is set to 10. We compared the DC-lasso proposed in [9] that is a re-weighted ℓ_1 scheme, the GIST [7] non-convex gradient descent method and our proposed active-set method using GIST for solving the inner problem (AS-GIST). All simulations were computed using Octave on Linux with a 2.60 GHz Intel processor and 64 GB of RAM[1].

Generated dataset. The dataset is generated as follows. A full matrix $\mathbf{A} \in \mathbb{R}^{n \times p}$ is drawn from the normal distribution and its columns are normalized to have unit norm. An extremely sparse model is generated with only t true active variables also drawn from a normal distribution. Finally an observation $\mathbf{y}$ is generated by adding Gaussian noise to the true model with a signal-to-noise ratio (SNR) of 30 dB. The regularization parameter λ is selected so as to promote the selection of approximately t variables in the estimated model. The results correspond to the average computational time obtained over 10 different generations of the data.

[1] Computed on `http://calculs.unice.fr/`

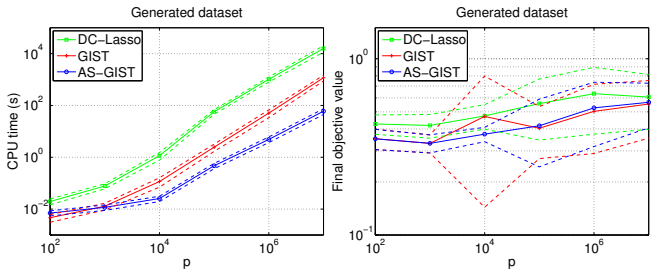

Fig. 1. CPU time and final objective value on the generated problem for a varying number of variables p. The standard deviation is also reported on the plot with the dashed lines

We computed in Figure 1 the CPU time necessary to obtain the solution of the optimization problem for $n = 100$, $t = 10$ and a number of variables p varying from 100 to 10^7. DC-Lasso is clearly outperformed both by GIST and AS-GIST. Moreover we can see that when $p \geq 1000$, AS-GIST is faster than GIST (up to 20 times for $p = 10^7$). A Wilcoxon sign test with $\alpha = 0.05$ was performed to compare the solutions obtained by the three algorithms. GIST and AS-GIST returned statistically equivalent objective values for all p but DC-Lasso was statistically worse than both methods. We believe that this can be explained by the different initialization used for both approaches, namely zero vector for GIST and AS-GIST and Lasso solution for DC-Lasso.

Real life datasets We evaluated our algorithm on two real life datasets available on the UCI Machine Learning Repository. First, the URL Reputation dataset [17] consists in several days of acquisition with $n = 20\,000$ examples per day with $p \approx 3.2 \times 10^6$ variables. Second, the Dorothea feature selection dataset [18] contains $n = 1150$ and $p = 10^5$. Both datasets are sparse and the relevant features are unknown. In these experiments, we run the GIST and AS-GIST algorithms with different values of the regularization parameter λ in order to see their performances with different sparsity ratios (DC-lasso was not considered here due to its limited performance and important computational burden). The URL dataset being naturally split in days, one model is learned from each of the first 10 days. On the Dorothea dataset, due to the small number of examples, 10 random splits of the data are drawn with 1000 out of 1150 training examples. The objective values, CPU time and number of selected features are averaged over the ten splits of data.

The performances in terms of CPU time and objective values are reported in Figure 2. We can see that AS-GIST is more CPU efficient than GIST for large values of λ, *i.e.* for sparser solutions (up to $500\times$ for the URL dataset). In addition, while both methods give equivalent solutions for large λ, the objective value of AS-GIST is clearly better for small λ. This interesting fact is the result of the ε thresholding in our algorithm, yielding sparser solutions than GIST and avoiding

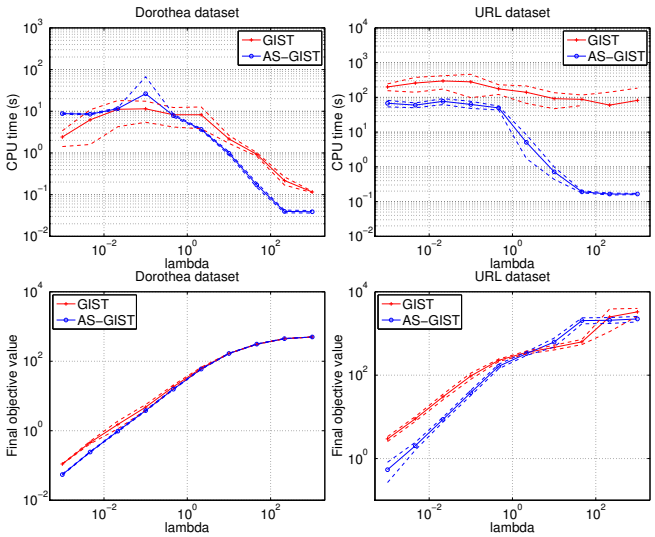

Fig. 2. CPU time (top) and objective value (bottom) for a varying regularization parameter λ on the Dorothea (left) and URL (right) datasets.

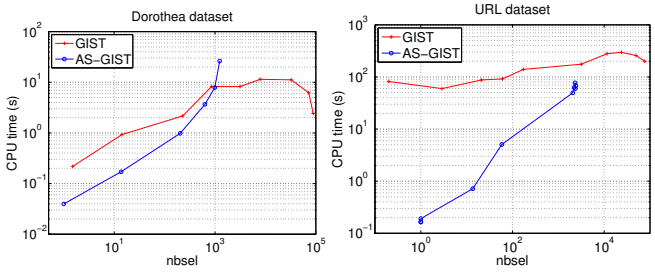

Fig. 3. CPU time as a function of the number of selected variables on the Dorothea (left) and URL (right) datasets.

spurious variables more often. Finally, the efficiency of the algorithms with respect to the number of selected variables is reported in Figure 3. It shows that, for a reasonable number of selected features (up to 1000), the active set strategy is extremely efficient, while the efficiency of GIST displays an almost constant behaviour with respect to the number of selected features.

5. CONCLUSION

We propose in this work an efficient active set strategy for non-convex sparse optimization. This approach sits on the shoulder of the existing non-convex optimization algorithms by improving their efficiency for very high dimensional problems. Numerical experiments show large computational gain of up to 2 orders of magnitude. As future works, this method can be extended to remote sensing and computer vision [19], for instance by applying non-convex optimization to feature selection [20].

6. REFERENCES

[1] D.L. Donoho, "De-noising by soft-thresholding," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 613–627, 1995.

[2] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[3] H. Zou, "The adaptive lasso and its oracle properties," *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1418–1429, 2006.

[4] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1348–1360, 2001.

[5] C.H. Zhang, "Nearly unbiased variable selection under minimax concave penalty," *Annals of Statistics*, vol. 38, no. 2, pp. 894–942, 2010.

[6] E.J. Candes, M.B. Wakin, and S.P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 877–905, 2008.

[7] P. Gong, C. Zhang, Z. Lu, and J. Huang, J. Ye, "A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems," in *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013, vol. 28, pp. 37–45.

[8] L.T.H. An and P.D. Tao, "The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems," *Annals of Operations Research*, vol. 133, no. 1, pp. 23–46, 2005.

[9] G. Gasso, A. Rakotomamonjy, and S. Canu, "Recovering sparse signals with a certain family of nonconvex penalties and DC programming," *IEEE Transactions on Signal Processing*, vol. 57, no. 12, pp. 4686–4698, 2009.

[10] Z. Lu, "Sequential convex programming methods for a class of structured nonlinear programming," *arXiv preprint arXiv:1210.3039*, 2012.

[11] J. Cheng, C. Hatzis, H. Hayashi, M-A. Krogel, S. Morishita, D. Page, and J. Sese, "KDD Cup 2001 report," *ACM SIGKDD Explorations Newsletter*, vol. 3, no. 2, pp. 47–64, 2002.

[12] M. Thiao, *Approches de la programmation DC et DCA en data mining: modélisation parcimonieuse de données*, Ph.D. thesis, INSA de Rouen, 2011.

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

[14] D.G. Luenberger, *Linear and nonlinear programming*, Springer, 2003.

[15] H. Zou and R. Li, "One-step sparse estimates in nonconcave penalized likelihood models," *Annals of Statistics*, vol. 36, no. 4, pp. 1509, 2008.

[16] S. Sra, "Nonconvex proximal splitting: batch and incremental algorithms," *Arxiv preprint arXiv:1109.0258*, 2011.

[17] J. Ma, L.K. Saul, S. Savage, and G.M. Voelker, "Identifying suspicious URLs: an application of large-scale online learning," in *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, 2009, pp. 681–688.

[18] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the NIPS 2003 feature selection challenge," in *Advances in Neural Information Processing Systems*, 2004, pp. 545–552.

[19] V. Krylov, G. Moser, S.B. Serpico, and J. Zerubia, "Supervised high resolution dual polarization SAR image classification by finite mixtures and copulas," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 3, pp. 554–566, June 2011.

[20] D. Tuia, R. Flamary, M. Volpi, M. Della Mura, and A. Rakotomamonjy, " Discovering relevant spatial filterbanks for VHR image classification ," in *International Conference on Pattern Recognition (ICPR)*, 2012.