DC Proximal Newton for Non-Convex Optimization Problems

Alain Rakotomamonjy, Remi Flamary, Gilles Gasso

▶ **To cite this version:**

Alain Rakotomamonjy, Remi Flamary, Gilles Gasso. DC Proximal Newton for Non-Convex Optimization Problems. 2014. hal-00952445v1

HAL Id: hal-00952445

https://hal.science/hal-00952445v1

Submitted on 26 Feb 2014 (v1), last revised 2 Jul 2015 (v3)

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.

DC Proximal Newton for Non-Convex Optimization Problems

A. Rakotomamonjy, R. Flamary, G. Gasso

***Abstract*—We introduce a novel algorithm for solving learning problems where both the loss function and the regularizer are non-convex but belong to the class of difference of convex (DC) functions. Our contribution is a new general purpose proximal Newton algorithm that is able to deal with such a situation. The algorithm consists in obtaining a descent direction from an approximation of the loss function and then in performing a line search to ensure sufficient descent. A theoretical analysis is provided showing that the iterates of the proposed algorithm admit as limit points stationary points of the DC objective function. Numerical experiments show that our approach is more efficient than current state of the art for a problem with a convex loss functions and non-convex regularizer. We have also illustrated the benefit of our algorithm in high-dimensional transductive learning problem where both loss function and regularizers are non-convex.**

***Index Terms*—Difference of convex functions, non-convex regularization, proximal Newton, sparse logistic regression.**

I. Introduction

In many real-world application domains such as computational biology, finance or text mining, datasets considered for learning prediction models are routinely large-scale and high-dimensional raising the issue of model complexity control. One way for dealing with such kinds of dataset is to learn sparse models. Hence, a very large amount of recent works in machine learning, statistics and signal processing have addressed optimization problems related to sparsity issues.

One of the most popular algorithm for achieving sparse models is the Lasso algorithm [1] also known as the Basis pursuit algorithm [2] in the signal processing community. This algorithm actually applies ℓ_1-norm regularization to the learning model. The choice of the ℓ_1 norm comes from its appealing properties which are convexity, continuity and its ability to produce sparse or even the sparsest model in some cases [3], [4]. Since these seminal works, several efforts have been devoted to the development of efficient algorithms for solving learning problems that consider sparsity-inducing regularizers [5]–[8]. However, ℓ_1 regularizer presents some drawbacks such as its inability, in certain situations to retrieve the true relevant variables of a model [9], [10]. Since the ℓ_1-norm regularizer is a continuous and convex surrogate of the ℓ_0 pseudo-norm, others kinds of regularizer, which relax the convexity property, have been analyzed by several authors and they have been proved to achieve better statistical property. Common non-convex regularizers are the SCAD regularizer [10], the ℓ_p regularizer [11], the capped-ℓ_1 and the log penalty [12]. These regularizers have been frequently used for feature selections or for obtaining sparse models [12]–[14].

While being statistically appealing, the use of these non-convex regularizers poses some challenging optimization problems and in this work, we propose a novel non-convex proximal Newton algorithm for solving such problems. Indeed, one of the most frequently used algorithms for solving ℓ_1-norm regularized problem is the proximal gradient algorithm [15]. Recently, proximal Newton-type methods have been introduced for solving composite optimization problems involving the sum of a smooth and convex twice differentiable function and a non-smooth convex function (typically the regularizer) [16], [17]. These proximal Newton algorithms have been shown to be substantially faster than their proximal gradient counterpart.

Based on this, we propose a general proximal Newton method for optimizing a composite objective function $f(\mathbf{x}) + h(\mathbf{x})$ where both functions f and h can be potentially non-convex and belong to a large class of functions that can be decomposed as the difference of two convex functions (DC functions) [18]–[20]. The proposed algorithm has a wide range of applicability that goes far beyond the handling of non-convex regularizers. Indeed, our global framework can genuinely deal with non-convex loss functions that usually appear in learning problems. To make concrete the DC Newton proximal approach, we illustrate the relevance and the effectiveness of the novel algorithm by considering a problem of sparse transductive logistic regression in which the regularizer as well as the loss related to the unlabeled examples are non-convex. As far as our knowledge goes, this is the first work that introduces such a model and proposes an algorithm for solving the related optimization problem.

Similar to convex proximal Newton method, the algorithm we propose consists in two steps: first it seeks a search direction and then it looks for a step-size in that direction that minimizes the objective value. The originality and main novelty we brought in this work are that the search direction is obtained by solving a subproblem which involves both an approximation of the smooth loss function and the DC regularizer. We prove several properties related to the obtained search direction and provide convergence analysis of the algorithm to a stationary point of the related optimization problem. Experimental studies show the benefit of the algo-

This work has been partly supported by the French ANR (09-EMER-001, 12-BS02-004 and 12-BS03-003).

AR is with LITIS EA 4108, Université de Rouen, France. alain.rakoto@insa-rouen.fr
RF is with Lagrange laroratory, Université de Nice Sophia-Antipolis, CNRS, Observatoire de la Côte d'Azur, F-06304 Nice, France. remi.flamary@unice.fr
GG is with LITIS EA 4108, INSA de Rouen, France. gilles.gasso@insa-rouen.fr

rithm in terms of running time while preserving or improving generalization performance compared to existing non-convex approaches.

II. DC PROXIMAL NEWTON ALGORITHM

We are interested in solving the following optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x}) := f(\mathbf{x}) + h(\mathbf{x}) \tag{1}$$

with the following assumptions concerning the functions f and h. f is supposed to be twice differentiable, lower bounded and we suppose that there exists two convex functions f_1 and f_2 such that $f(\mathbf{x}) = f_1(\mathbf{x}) - f_2(\mathbf{x})$. We also assume that f_1 verifies the L-Lipschitz gradient property

$$\|\nabla f_1(\mathbf{x}) - \nabla f_1(\mathbf{y})\| \leq L\|\mathbf{x}-\mathbf{y}\| \quad \forall \mathbf{x},\mathbf{y} \in \mathrm{dom} f_1.$$

The function h is supposed to be a lower-bounded, proper lower semi-continuous function that can also be expressed as

$$h(\mathbf{x}) = h_1(\mathbf{x}) - h_2(\mathbf{x}) \tag{2}$$

where h_1 and h_2 are both convex functions. As discussed in the introduction, we focus our interest in situations where at least h is non-convex.

A. Optimization scheme

For solving Problem (1) which is a difference of convex functions optimization problem, we propose a novel iterative algorithm which first looks for a search direction $\Delta\mathbf{x}$ and then updates the current solution. Formally, the algorithm is based on the iteration

$$\mathbf{x}_{k+1} = \mathbf{x}_k + t_k \Delta\mathbf{x}_k$$

where t_k and $\Delta\mathbf{x}_k$ are respectively a step size and the search direction. Similarly to the works of Lee et al. [16], the search direction is computed by minimizing a local approximation of the composite function $F(\mathbf{x})$. However, we show that by using a simple approximation on f_1, f_2 and h_2, we are able to handle the non-convexity of $F(\mathbf{x})$, resulting in an algorithm which is wrapped around a specific proximal Newton iteration.

For dealing with the non-convex situation, we define the search direction as the solution of the following problem

$$\Delta\mathbf{x}_k = \arg\min_{\Delta x} \tilde{f}(\mathbf{x}_k + \Delta\mathbf{x}) + \tilde{h}(\mathbf{x}_k + \Delta\mathbf{x}) \tag{3}$$

where $\tilde{f}$ and $\tilde{h}$ are the following approximations of respectively f and h at $\mathbf{x}_k$. We define $\tilde{f}(\mathbf{x})$ as

$$\begin{aligned}\tilde{f}(\mathbf{x}) &= f_1(\mathbf{x}_k) + \nabla f_1(\mathbf{x}_k)^\top(\mathbf{x}-\mathbf{x}_k) \\ &+ \frac{1}{2}(\mathbf{x}-\mathbf{x}_k)^\top \mathbf{H}_k(\mathbf{x}-\mathbf{x}_k) \\ &- f_2(\mathbf{x}_k) - \mathbf{z}_{f_2}^\top(\mathbf{x}-\mathbf{x}_k)\end{aligned} \tag{4}$$

where $\mathbf{z}_{f_2} = \nabla f_2(\mathbf{x}_k)$ and $\mathbf{H}_k$ is any positive definite approximation of the Hessian matrix of f_1 at current iterate. We also consider

$$\tilde{h}(\mathbf{x}) = h_1(\mathbf{x}) - h_2(\mathbf{x}_k) - \mathbf{z}_{h_2}^\top(\mathbf{x}-\mathbf{x}_k) \tag{5}$$

where $\mathbf{z}_{h_2} \in \partial h_2(\mathbf{x}_k)$, with the latter being the sub-differential of h_2 at $\mathbf{x}_k$.

Algorithm 1 DC proximal Newton algorithm

```
Initialize x_0 ∈ domF
k = 0
repeat
    compute z_{h_2} ∈ ∂h_2(x_k) and z_{f_2} = ∇f_2(x_k)
    update H_k (exactly or using a quasi-Newton approach)
    v_k ← ∇f_1(x_k) − z_{f_2} − z_{h_2}
    Δx_k ← prox^{H_k}_{h_1}(x_k − H_k^{-1} v_k) − x_k
    compute the stepsize t_k through backtracking
    x_{k+1} = x_k + t_k Δx_k
    k ← k + 1
until convergence criterion is met
```

Note that the first three summands in Equation (4) form a quadratic approximation of $f_1(\mathbf{x})$ whereas the terms in the third line of Equation (4) is a linear approximation of $f_2(\mathbf{x})$. In the same spirit, $\tilde{h}$ is actually a majorizing function of h since we have linearized the convex function h_2 and h is a difference of convex functions.

We are now in position to provide the proximal expression of the search direction. Indeed, Problem (3) can be rewritten as

$$\arg\min_{\Delta\mathbf{x}} \frac{1}{2}\Delta\mathbf{x}^\top\mathbf{H}_k\Delta\mathbf{x} + h_1(\mathbf{x}_k+\Delta\mathbf{x}) + \mathbf{v}_k{}^\top\Delta\mathbf{x} \tag{6}$$

with $\mathbf{v}_k = \nabla f_1(\mathbf{x}_k) - \mathbf{z}_{f_2} - \mathbf{z}_{h_2}$ and after some algebras involving optimality conditions of a proximal Newton operator, we can show that

$$\Delta\mathbf{x}_k = \mathbf{prox}_{h_1}^{\mathbf{H}_k}(\mathbf{x}_k - \mathbf{H}_k^{-1}\mathbf{v}_k) - \mathbf{x}_k \tag{7}$$

with by definition [15], [16]

$$\mathbf{prox}_{h_1}^{\mathbf{H}}(\mathbf{x}) = \arg\min_{\mathbf{y}} \frac{1}{2}\|\mathbf{x}-\mathbf{y}\|_{\mathbf{H}}^2 + h_1(\mathbf{y})$$

Interestingly, we note that the non-convexity of the initial problem is taken into account only through the proximal Newton operator and its impact on the algorithm, compared to the convex case, is minor since it only modifies the argument of the operator through $\mathbf{v}_k$.

Once the search direction is computed, the step size t_k is backtracked starting from 1. Algorithm 1 summarizes the main steps of the optimization scheme. Some implementation issues are discussed hereafter while the next section focuses on the convergence analysis.

B. Implementation's tricks of the trade

The main difficulty and computational burden of our DC proximal Newton algorithm resides in the computation of the search direction $\Delta\mathbf{x}_k$. Indeed, the latter needs the computation of the proximal operator $\mathbf{prox}_{h_1}^{\mathbf{H}_k}(\mathbf{x}_k - \mathbf{H}_k^{-1}\mathbf{v}_k)$ which is equal to

$$\arg\min_{\mathbf{y}} \underbrace{\frac{1}{2}\mathbf{y}^\top\mathbf{H}_k\mathbf{y} + \mathbf{y}^\top(\mathbf{v}_k - \mathbf{H}_k\mathbf{x}_k)}_{g(\mathbf{y})} + h_1(\mathbf{y}) \tag{8}$$

We can note that this problem is a quadratic problem penalized by h_1. If $h_1(\mathbf{y})$ is a term which proximal operator

can be cheaply computed then, one can consider proximal gradient algorithm or any other efficient algorithms for its resolution [6], [21].

In our case, we have considered a forward-backward (FB) algorithm [15] initialized with the previous value of the optimal $\mathbf{y}$. Note that in order to have a convergence guarantee, the FB algorithm needs a stepsize smaller than $\frac{2}{L}$ where L is the Lipschitz gradient of the quadratic function. Again computing L can be expensive and in order to increase the computational efficiency of the global algorithm, we have chosen a strategy that roughly estimates L according to the equation

$$\frac{\|\nabla g(\mathbf{y}) - \nabla g(\mathbf{y}')\|_2}{\|\mathbf{y} - \mathbf{y}'\|_2}$$

In practice, we have found this heuristic to be slightly more efficient than an approach which computes the largest eigenvalue of $\mathbf{H}_k$ by means of a power method [22]. Note that a L-BFGS approximation scheme has been used in the numerical experiments for updating the matrix $\mathbf{H}_k$.

While the convergence analysis we provide in the next section supposes that the proximal operator is computed exactly, in practice it is more efficient to approximately solve the search direction problem, at least for the early iterations. Following this idea, we have considered an adaptive stopping criterion for the proximal operator subproblem.

C. Related works

In the last few years, a large amount of works have been devoted to the resolution of composite optimization problem of the form given in Equation (1). We review the ones that are most similar to ours.

Proximal Newton algorithms have recently been proposed by [16] and [17] for solving Equation (1) when both functions $f(\mathbf{x})$ and $h(\mathbf{x})$ are convex. While the algorithm we propose is similar to the one of [16], our work is strictly more general in the sense that we relax the convexity hypothesis on both functions. Indeed, our algorithm can handle both convex and non-convex cases and boils down to the algorithm of [16] in the convex case.

Following the interest on sparsity-inducing regularizers, there has been a renewal of curiosity around non-convex optimization problems [12], [13]. Indeed, most statistically relevant sparsity-inducing regularizers are non-convex [23]. Hence, several researchers have proposed novel algorithms for handling these isssues.

We also notice that linearizing the concave part in a DC program is a crucial idea of DC programming and DCA that were introduced by Pham Dinh Tao in the early eighties and have been extensively developed since then [18], [19], [24]. In this work, we have used this same idea in a proximal Newton framework. However, our algorithm is fairly different from the DCA [19] as we consider a single descent step at each iteration, as opposed to the DCA which needs a full optimization of a minimization problem at each iteration. This idea of linearizing the (possibly) non-convex part of Problem (1) for obtaining a search direction can also be found in Mine et al. [25]. However, in their case, the function to be linearized is supposed to be smooth. The advantage of using a DC program as in our case is that, the linearization trick can also be extended to non-smooth function.

The works that are mostly related to ours are those proposed by [26] and [27]. Interestingly, [26] introduced an iterative shrinkage algorithm that can handle optimization problems with DC regularizers for which proximal operators can be easily computed. Instead, [27] solves the same optimization problem in a different way. As the non-convex regularizers are supposed to be DC, they proposed to solve a sequence of convex programs which at each iteration minimizes

$$\tilde{f}(\mathbf{x}) + h_1(\mathbf{x}) - h_2(\mathbf{x}_k) - \mathbf{z}_{h_2}^\top(\mathbf{x} - \mathbf{x}_k)$$

with

$$\tilde{f}(\mathbf{x}) = f_1(\mathbf{x}_k) + \nabla f_1(\mathbf{x}_k)^\top(\mathbf{x} - \mathbf{x}_k) + \frac{1}{2}\|\mathbf{x} - \mathbf{x}_k\|^2$$

Note that our framework subsumes the one of Lu [27] in several points. Indeed, the algorithm we propose can also handle non-convexity over the differentiable term and we take into account a variable metric $\mathbf{H}_k$ into the proximal term. Thus, the approach of Lu can be deemed a particular case of our method when $f_2(\mathbf{x}) = 0$ in Problem (1) and $\mathbf{H}_k = \mathbf{I}$ at all iterations of the algorithm. Hence, when $f(\mathbf{x})$ is convex, we expect more efficiency than the algorithm of [26] and [27] owing to the variable metric $\mathbf{H}_k$ that has been introduced.

Very recently, Chouzenoux et al. [28] introduced a proximal Newton-like algorithm for minimizing the sum of a twice differentiable function and a convex function. They essentially consider that the regularization term is convex while the loss function may be non-convex. Their work can thus be seen as an extension of the one of [29] to the variable metric $\mathbf{H}_k$ case. Compared to our work, [28] do not impose a DC condition on the function $f(\mathbf{x})$. However, at each iteration, they need a quadratic surrogate function at a point $\mathbf{x}_k$ that majorizes $f(\mathbf{x})$. In our case, only the non-convex part is majorized through a simple linearization.

Again, we stress that our framework is more general than all above presented works as none of them considered cases where both composite terms $f(\mathbf{x})$ and $h(\mathbf{x})$ are non-convex. In addition, we show that a proximal Newton algorithm can be developed in such a context by means of a DC approach.

III. Analysis

Our objective in this section is to show that our algorithm is well-behaved and prove that the iterates $\{\mathbf{x}_k\}$ converge to a stationary point of Problem (1). We first characterize stationary points of Problem 1 with respects to $\Delta\mathbf{x}$ and then show that all limit points of the sequence $\{\mathbf{x}_k\}$ generated by our algorithm are stationary points.

Throughout this work, we use the following definition of a stationary point.

Definition 1: A point $\mathbf{x}^\star$ is said to be a stationary point of Problem (1) if

$$0 \in \nabla f_1(\mathbf{x}^\star) - \nabla f_2(\mathbf{x}^\star) + \partial h_1(\mathbf{x}^\star) - \partial h_2(\mathbf{x}^\star)$$

According to the above definition, we have the following lemma :

Lemma 1: Suppose $\mathbf{H}_\star \succ 0$, $\mathbf{x}^\star$ is a stationary point of Problem (1) if and only if $\Delta\mathbf{x}^\star = 0$ with

$$\Delta\mathbf{x}^\star = \arg\min_{\mathbf{d}} (\mathbf{v}^\star)^\top \mathbf{d} + \frac{1}{2}\mathbf{d}^\top \mathbf{H}_\star \mathbf{d} + h_1(\mathbf{x}^\star + \mathbf{d}) \quad (9)$$

and $\mathbf{v}^\star = \nabla f_1(\mathbf{x}^\star) - \mathbf{z}_{f_2}^\star - \mathbf{z}_{h_2}^\star$, $\mathbf{z}_{f_2}^\star = \nabla f_2(\mathbf{x}^\star)$ and $\mathbf{z}_{h_2}^\star \in \partial h_2(\mathbf{x}^\star)$.

Proof : Let us start by characterizing the solution $\Delta\mathbf{x}^\star$. By definition, we have $\Delta\mathbf{x}^\star + \mathbf{x}^\star = \mathbf{prox}_{h_1}^{\mathbf{H}_\star}(\mathbf{x}^\star - \mathbf{H}_\star^{-1}\mathbf{v}^\star)$ and thus according to the optimality condition of the proximal operator, the following equation holds

$$\mathbf{H}_\star(\mathbf{x}^\star - \mathbf{H}_\star^{-1}\mathbf{v}^\star - \Delta\mathbf{x}^\star - \mathbf{x}^\star) \in \partial h_1(\Delta\mathbf{x}^\star + \mathbf{x}^\star)$$

which after rearrangement is equivalent to

$$\mathbf{z}_{h_2}^\star - \mathbf{H}_\star\Delta\mathbf{x}^\star \in \nabla f(\mathbf{x}^\star) + \partial h_1(\Delta\mathbf{x}^\star + \mathbf{x}^\star) \quad (10)$$

with $\nabla f(\mathbf{x}^\star) = \nabla f_1(\mathbf{x}^\star) - \nabla f_2(\mathbf{x}^\star)$. This also means that there exist a $\mathbf{z}_{h_1\Delta}^\star \in \partial h_1(\Delta\mathbf{x}^\star + \mathbf{x}^\star)$ so that

$$\mathbf{z}_{h_2}^\star - \mathbf{H}_\star\Delta\mathbf{x}^\star - \nabla f(\mathbf{x}^\star) - \mathbf{z}_{h_1\Delta}^\star = \mathbf{0} \quad (11)$$

Remember that by hypothesis, since $\mathbf{x}^\star$ is a stationary point of Problem (1), we have

$$\mathbf{0} \in \nabla f(\mathbf{x}^\star) + \partial h_1(\mathbf{x}^\star) - \partial h_2(\mathbf{x}^\star)$$

We now prove that if $\mathbf{x}^\star$ is a stationary point of Problem (1) then $\Delta\mathbf{x} = \mathbf{0}$ by showing the contrapositive. Suppose that $\Delta\mathbf{x}^\star \neq \mathbf{0}$. $\Delta\mathbf{x}^\star$ is a vector that satisfies the optimality condition (10) and it is the unique one according to properties of the proximal operator. This means that the vector $\mathbf{0}$ is not optimal for the problem (9) and thus it does not exist a vector $\mathbf{z}_{h_1\mathbf{0}}^\star \in \partial h_1(\mathbf{d} + \mathbf{x}^\star)$ so that

$$\mathbf{z}_{h_2}^\star - \mathbf{H}_\star\mathbf{d} - \nabla f(\mathbf{x}^\star) - \mathbf{z}_{h_1\mathbf{0}}^\star = \mathbf{0} \quad (12)$$

with $\mathbf{d} = \mathbf{0}$. Note that this equation is valid for any $\mathbf{z}_{h_2}^\star$ chosen in the set $\partial h_2(\mathbf{x}^\star)$ and the above equation also translates in $\nexists, \mathbf{z}_{h_1\mathbf{0}}^\star \in \partial h_1(\mathbf{x}^\star)$ so that $\nabla f(\mathbf{x}^\star) + \mathbf{z}_{h_1\mathbf{0}}^\star - \mathbf{z}_{h_2}^\star = \mathbf{0}$, which proves that $\mathbf{x}^\star$ is not a stationary point of problem (1).

Suppose now that $\Delta\mathbf{x}^\star = 0$, then according to the definition of $\Delta\mathbf{x}^\star$ and the resulting condition (10), it is straightforward to note that $\mathbf{x}^\star$ satisfies the definition of a stationary point. □

Now, we proceed by showing that at each iteration, the search direction $\Delta\mathbf{x}_k$ satisfies a property which implies that for a sufficiently small step size t_k, the search direction is a descent direction.

Lemma 2: For $\mathbf{x}_k$ in the domain of f and supposing that $\mathbf{H}_k \succ 0$ then $\Delta\mathbf{x}_k$ is so that

$$F(\mathbf{x}_{k+1}) \leq F(\mathbf{x}_k) + t_k\Big(\mathbf{v}_k^\top\Delta\mathbf{x}_k + h_1(\Delta\mathbf{x}_k + \mathbf{x}_k) - h_1(\mathbf{x}_k)\Big) + O(t_k^2)$$

and

$$F(\mathbf{x}_{k+1}) - F(\mathbf{x}_k) \leq -t_k\Delta\mathbf{x}_k^\top\mathbf{H}_k\Delta\mathbf{x}_k + O(t_k^2) \quad (13)$$

with $\mathbf{v}_k = \nabla f_1(\mathbf{x}_k) - \mathbf{z}_{f_2} - \mathbf{z}_{h_2}$.

Proof: For a sake of clarity, we have dropped the index k and used the following notation. $\mathbf{x} := \mathbf{x}_k$, $\Delta\mathbf{x} := \Delta\mathbf{x}_k$, $\mathbf{x}_+ := \mathbf{x}_k + t_k\Delta\mathbf{x}_k$. By definition, we have

$$\begin{aligned} F(\mathbf{x}_+) - F(\mathbf{x}) = {} & f_1(\mathbf{x}_+) - f_1(\mathbf{x}) - f_2(\mathbf{x}_+) + f_2(\mathbf{x}) \\ & + h_1(\mathbf{x}_+) - h_1(\mathbf{x}) - h_2(\mathbf{x}_+) + h_2(\mathbf{x}). \end{aligned}$$

Then by convexity of f_2, h_2, h_1 and for $t \in [0,1]$, we respectively have

$$\begin{aligned} -\mathbf{z}_{f_2}^\top(\mathbf{x}_+ - \mathbf{x}) &\geq f_2(\mathbf{x}) - f_2(\mathbf{x}_+), \\ -\mathbf{z}_{h_2}^\top(\mathbf{x}_+ - \mathbf{x}) &\geq h_2(\mathbf{x}) - h_2(\mathbf{x}_+) \end{aligned}$$

and

$$h_1(\mathbf{x} + t\Delta\mathbf{x}) \leq t h_1(\mathbf{x} + \Delta\mathbf{x}) + (1-t)h_1(\mathbf{x})$$

Plugging these inequalities in the definition of $F(\mathbf{x}_+) - F(\mathbf{x})$ gives :

$$\begin{aligned} F(\mathbf{x}_+) - F(\mathbf{x}) &\leq f_1(\mathbf{x}_+) - f_1(\mathbf{x}) + (1-t)h_1(\mathbf{x}) \\ &\quad + t h_1(\mathbf{x} + \Delta\mathbf{x}) \\ &\quad - t(\mathbf{z}_{f_2} + \mathbf{z}_{h_2})^\top\Delta\mathbf{x} - h_1(\mathbf{x}) \\ &\leq t\nabla f_1(\mathbf{x})^\top\Delta\mathbf{x} + t h_1(\mathbf{x} + \Delta\mathbf{x}) \\ &\quad - t h_1(\mathbf{x}) - t(\mathbf{z}_{f_2} + \mathbf{z}_{h_2})^\top\Delta\mathbf{x} + O(t^2) \end{aligned} \quad (14)$$

which proves the first inequality of the lemma.

For showing the descent property, we demonstrate that the following inequality holds

$$\underbrace{\mathbf{v}^\top\Delta\mathbf{x} + h_1(\mathbf{x} + \Delta\mathbf{x}) - h_1(\mathbf{x})}_{D} \leq -\Delta\mathbf{x}^\top\mathbf{H}\Delta\mathbf{x} \quad (15)$$

Since $\Delta\mathbf{x}$ is the minimizer of Problem (6), the following equation holds for $t\Delta\mathbf{x}$ and $t \in [0,1]$:

$$\begin{aligned} &\frac{1}{2}\Delta\mathbf{x}^\top\mathbf{H}\Delta\mathbf{x} + h_1(\mathbf{x} + \Delta\mathbf{x}) + \mathbf{v}^\top\Delta\mathbf{x} \\ &\leq \frac{t^2}{2}\Delta\mathbf{x}^\top\mathbf{H}\Delta\mathbf{x} + h_1(\mathbf{x} + t\Delta\mathbf{x}) + t\mathbf{v}^\top\Delta\mathbf{x} \\ &\leq \frac{t^2}{2}\Delta\mathbf{x}^\top\mathbf{H}\Delta\mathbf{x} + (1-t)h_1(\mathbf{x}) + t h_1(\mathbf{x} + t\Delta\mathbf{x}) \\ &\quad + t_k\mathbf{v}^\top\Delta\mathbf{x} \end{aligned} \quad (16)$$

After rearrangement we have the inequality

$$\mathbf{v}_k^\top\Delta\mathbf{x}_k + h_1(\mathbf{x}_k + \Delta\mathbf{x}_k) - h_1(\mathbf{x}_k) \leq -\frac{1}{2}(1+t_k)\Delta\mathbf{x}_k^\top\mathbf{H}_k\Delta\mathbf{x}_k$$

which is valid for all $t_k \in [0,1]$ and in particular for $t_k = 1$ which concludes the proof of inequality. By plugging this result into inequality (14), the descent property holds. □

Note that the descent property is supposed to hold for sufficiently small step size. In our algorithm, this stepsize t_k is selected by backtracking so that the following sufficient descent condition holds

$$F(\mathbf{x}_{k+1}) - F(\mathbf{x}_k) \leq \alpha t_k D_k \quad (17)$$

with $\alpha \in (0, 1/2)$. The next lemma shows that if the function f_1 is sufficiently smooth, then there always exists a step size so that the above sufficient descent condition holds.

Lemma 3: For $\mathbf{x}$ in the domain of f and assuming that $\mathbf{H}_k \succeq m\mathbf{I}$ with $m > 0$ and ∇f_1 is Lipschitz with constant L then the sufficient descent condition in Equation (17) holds for all t_k so that

$$t_k \leq \min\left(1, 2m\frac{1-\alpha}{L}\right)$$

Proof : This technical proof has been post-poned to the appendix. □

According to the above lemma, we can suppose that if some mild conditions on f_1 are satisfied (smoothness and bounded curvature) then, we can expect our DC algorithm to behave properly. This intuition is formalized in the following property

Proposition 1: Suppose f_1 has a gradient which is Lipschitz continuous with constant L and that $\mathbf{H}_k \succeq m\mathbf{I}$ for all k and $m > 0$, then all the limit points of the sequence $\{\mathbf{x}_k\}$ are stationary points.

Proof : Let $\mathbf{x}^\star$ be a limit point of the sequence $\{\mathbf{x}_k\}$ then, there exists a subsequence $\mathcal{K}$ so that

$$\lim_{k\to\mathcal{K}} \mathbf{x}_k = \mathbf{x}^\star$$

At each iteration the step size t_k has been chosen so as to satisfy the sufficient descent condition given in Equation (17). According to the above Lemma 3, the step size t_k is chosen so as to ensure a sufficient descent and we know that such a step size always exists and it is always non-zero. Hence the sequence $\{F(\mathbf{x}_k)\}$ is a strictly decreasing sequence. As F is lower bounded, the sequence $\{F(\mathbf{x}_k)\}$ converges to some limit. Thus, we have

$$\lim_{k\to\infty} F(\mathbf{x}_k) = \lim_{k\to\mathcal{K}} F(\mathbf{x}_k) = F(\mathbf{x}^\star)$$

as $F(\cdot)$ is continuous. Thus, we also have

$$\lim_{k\to\mathcal{K}} F(\mathbf{x}_{k+1}) - F(\mathbf{x}_k) = 0$$

and because each term $F(\mathbf{x}_{k+1}) - F(\mathbf{x}_k)$ is negative from Equation (17), we can also deduce that

$$\lim_{k\to\mathcal{K}} \mathbf{v}_k^\top \Delta\mathbf{x}_k + h_1(\mathbf{x}_k+\Delta\mathbf{x}_k) - h_1(\mathbf{x}_k) = \lim_{k\to\mathcal{K}} -\Delta\mathbf{x}_k^\top \mathbf{H}_k \Delta\mathbf{x}_k = 0$$

which from Equation (15) also means that

$$\lim_{k\to\mathcal{K}} \Delta\mathbf{x}_k = 0$$

since H_k is definite positive. Considering now that $\Delta\mathbf{x}_k$ is a minimizer of Problem (6), we have

$$\mathbf{0} \in \mathbf{H}_k\Delta\mathbf{x}_k + \partial h_1(\mathbf{x}_k+\Delta\mathbf{x}_k) + \nabla f_1(\mathbf{x}_k) - \nabla f_2(\mathbf{x}_k) - \partial h_2(\mathbf{x}_k)$$

Now, by taking limits on both side of the above equation for $k \in \mathcal{K}$, we have

$$\mathbf{0} \in \partial h_1(\mathbf{x}^\star) + \nabla f_1(\mathbf{x}^\star) - \partial h_2(\mathbf{x}^\star) - \nabla f_2(\mathbf{x}^\star)$$

Thus, $\mathbf{x}^\star$ is a stationary point of Problem (1). □

IV. Experiments

In order to provide evidence on the benefits of the proposed approach for solving DC non-convex problems, we have carried out two numerical experiments.

First we analyze our algorithm when the function f is convex and the regularizer h is a non-convex sparsity inducing penalty. Second, we study the case when both f and h are non-convex.

A. Sparse Logistic Regression

We are considering here that the loss function f is convex and is written as

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \log(1 + \exp(-y_i \mathbf{a}_i^\top \mathbf{x})$$

where $\{\mathbf{a}_i, y_i\}_{i=1}^{\ell}$ are the training examples and their associated labels available for learning the model. The regularizer we have considered is the capped-ℓ_1 defined as $h(\mathbf{x}) = h_1(\mathbf{x}) - h_2(\mathbf{x})$ with

$$h_1(\mathbf{x}) = \lambda\|\mathbf{x}\|_1 \text{ and } h_2(\mathbf{x}) = \lambda\big(\|\mathbf{x}\|_1 - \theta\big)_+ \qquad (18)$$

and the operator $(u)_+ = u$ if $u \geq 0$ and 0 otherwise.

Since several other algorithms are able to solve the optimization problem related to this sparse logistic regression problem, our objective here is to show that the proposed DC proximal Newton is computationally more efficient than competitors, while achieving equivalent classification performances. For this experiment, we have considered a single competitor which is the recently proposed GIST algorithm [26]. Indeed, this latter approach has already been shown by the authors to be more efficient than several other competitors including SCP (sequential convex programming) [27], MultiStage Sparsa [30]. Note that a slight advantage has been provided to GIST as we consider its non-monotone version (more efficient than the monotone counterpart) whereas our approach decreases the objective value at each iteration. DC Algorithm as described in section II-C has not been compared to our approach in those experiments as it has already been shown to be less efficient than GIST in [31].

1) Toy dataset: We have firstly evaluated GIST and our DC proximal Newton on a toy dataset where only few features are relevant for the discrimination task.

The toy problem is the same as the one used by [32]. The task is a binary classification problem in $\mathbb{R}^d$. Among these d variables, only T of them define a subspace of $\mathbb{R}^d$ in which classes can be discriminated. For these T relevant variables, the two classes follow a Gaussian pdf with means respectively μ and $-\mu$ and covariance matrices randomly drawn from a Wishart distribution. μ has been randomly drawn from $\{-1, +1\}^r$. The other $d-r$ non-relevant variables follow an *i.i.d* Gaussian probability distribution with zero mean and unit variance for both classes. We have respectively sampled N, and $n_t = 5000$ number of examples for training and testing. Before learning, the training set has been normalized to zero mean and unit variance and test set has been rescaled accordingly. Note that the hyperparameters λ and θ of the

regularization term (18) have been set so as to maximize the performance of the GIST algorithm on the test set. We have chosen to initialize all algorithms with zero vector ($\mathbf{x}^0 = \mathbf{0}$) and we terminate them if the relative change of the two consecutive objective function values is less than 10^{-6}.

Reported performances and running times averaged over 10 trials are depicted in Table I for two different settings of the dimensionality d and the number of training examples N. We note that for both problems our DC proximal Newton is computationally less demanding than GIST while the recognition performances are equivalent. Interestingly, we can remark that both algorithms yield to nearly similar objective values.

2) Benchmark datasets: The same experiments and protocols have been carried out on real-world high-dimensional learning problems. These datasets are those already used by [26] for illustrasting the behaviour of their GIST algorithm.

From Table II, we can note that while almost equivalent, recognition performances are sometimes statistically better for one method than the other although there is no clear winner. From the running time point of view, our DC proximal Newton exhibits a better behaviour. Indeed, its running time is always better, regardless of the dataset, and the difference in performance is statistically significantly better for 3 out of 5 datasets. In addition, we can note that in some situations, the gain in running time reaches an order of magnitude, clearly showing the benefit of a proximal Newton approach.

B. Sparse Transductive Logistic Regression

In this other experiment, we show an example of situation where one has to deal with a non-convex loss function as well as a non-convex regularizer, namely : sparse transductive logistic regression. The principle of transductive learning is to use unlabeled examples during the training step. This is usually done by using a loss function penalizing examples in the margin of the classifier. It is well known that this approach, also known as low density separation, leads to non-convex data fitting term on the unlabeled examples. For instance, Joachims et al. have considered a Symmetric Hinge loss for the unlabeled examples in their transductive implementation of SVM [33]. More recently, Collobert et al. [34] extended this idea of symmetric Hinge loss into a symmetric ramp loss, which has a plateau on its top. In order to have a smooth transductive loss, Chapelle et al [35] used a symmetric sigmoid loss.

In our case, since we also need that the transductive loss function to be differentiable, we propose the following symmetric differentiable loss that can be written as a difference of convex function

$$T(u) = 1 - g_1(u) - g_2(u)$$

where $g(u) = \log(1+\exp(-u))$, $g_1(u) = \frac{1}{\tau}(g(u) - g(u+\tau))$ and $g_2(u) = g_1(-u)$. Here, τ is a parameter that modifies the smoothness of $T(\cdot)$. From the expression of g_1 and g_2, it is easy to retrieve the difference of convex functions form of $T(\cdot)$. The transductive loss $T(\cdot)$ as well as g_1 and g_2 and their components are illustrated in Figure 1.

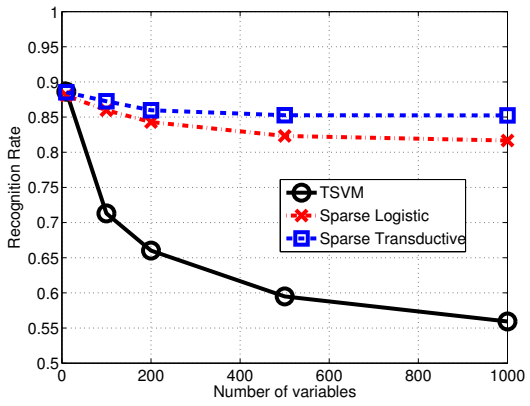

Figure 2. Recognition rate of different algorithms that are either sparse, transductive or both with respects to the number of variables in the problem, the number of relevant variables being 5.

According to this definition of the transductive loss, for our experiments, we have considered the following loss involving all training examples

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} g(y_i \mathbf{a}_i^\top \mathbf{x}) + \gamma \sum_{j=1}^{\ell_u} T(\mathbf{b}_j^\top \mathbf{x}) \qquad (19)$$

$\{\mathbf{a}_i, y_i\}$ being the labeled examples and $\{\mathbf{b}_j\}$ the unlabeled ones and γ is an hyperparameter that balances the weight of both losses.

1) Toy dataset: In order to illustrate the benefit of our sparse transductive approach, we have considered the same toy dataset as in the previous subsection and the same experimental protocol. However, we have considered only 5 relevant variables, sampled 100 training examples and 5000 testing examples. In addition, we have considered 10000 unlabeled examples. The total number of variables is varying. We have compared the recognition performance of 3 algorithms : the above-described capped-ℓ_1 sparse logistic regression, the non-sparse transductive SVM (TSVM) of [35][1] and our sparse transductive logistic regression.

Evolution of the recognition rate of these algorithms with respects to the number of variables in the learning problem is depicted in Figure 2. Interestingly, when the number of variables is small enough, all algorithms perform equivalently. Then, as the number of (noisy) variables increases, the transductive SVM suffers and loses performances. It seems more beneficial in this case to consider a model that is able to select relevant variables as our capped-ℓ_1 sparse logistic regression still performs good. Best performances are obtained using our sparse transductive logistic regression which is able to remove noisy variables and take advantage of the unlabeled examples.

2) Benchmark datasets: We have also analyzed the benefit of using unlabeled examples in high-dimensional learning problems. For this experiment, all the hyperparameters of all models have been cross-validated and average results over 10 trials are reported in Table III. Note that the results of the transductive SVM of [35] have not been reported because the

[1] we used the code available on the author's website.

d= 2000, N= 100000, $\lambda = 2.00$ $\theta = 0.20$					
	Class. Rate (%)		Time (s)		Obj Val (%)
T	GIST	DC-PN	GIST	DC-PN	Rel. Diff
50	91.86±1.5	91.83±1.5	62.32±10.6	**47.82±10.8**	-0.746
100	91.45±1.9	91.44±1.9	66.47±14.3	**46.28±10.3**	-0.000
500	91.54±0.6	91.51±0.6	69.59±4.5	59.41±19.9	-0.251
1000	91.64±0.7	91.64±0.7	75.57±7.4	**60.97±16.9**	-0.313

d= 10000, N= 5000, $\lambda = 2.00$ $\theta = 2.00$					
	Class. Rate (%)		Time (s)		Obj Val (%)
T	GIST	DC-PN	GIST	DC-PN	Rel. Diff
50	88.31±2.4	88.37±2.4	55.13±7.3	**31.44±2.0**	0.030
100	87.41±2.8	87.43±2.8	43.27±4.3	**29.79±3.0**	0.010
500	81.75±0.7	**81.83±0.7**	40.03±2.9	**26.05±0.5**	0.004
1000	76.66±0.9	76.69±0.9	39.09±4.4	**26.46±0.6**	0.014

Table I

COMPARISON WITH GIST AND OUR DC PROXIMAL NEWTON ON TOY PROBLEMS WITH INCREASING NUMBER OF RELEVANT VARIABLES. PERFORMANCES REPORTED IN **BOLD** ARE STATISTICALLY SIGNIFICANTLY DIFFERENT THAN THEIR COMPETITOR COUNTERPART ACCORDING TO A WILCOXON SIGNED RANK TEST WITH A P-VALUE AT 0.05. A MINUS SIGN IN THE RELATIVE OBJECTIVE VALUE INDICATES THAT THE DC PROXIMAL NEWTON APPROACH PROVIDES LARGER OBJECTIVE VALUE THAN GIST.

			Class. Rate (%)		Time (s)		Obj Val (%)
dataset	N	d	GIST	DC-PN	GIST	DC-PN	
la2	2460	31472	91.56±0.6	**92.13±0.6**	20.91±15.8	11.47±5.5	-94.356
sports	6864	14870	98.09±0.3	98.16±0.3	108.36±101.0	**11.62±5.9**	-30.606
classic	5675	41681	**96.40±0.5**	95.63±0.5	28.40±14.6	22.71±13.3	1.648
ohscal	8929	11465	87.72±0.5	88.96±0.5	48.82±17.2	**17.93±18.3**	-64.411
real-sim	57847	20958	**96.26±0.2**	96.05±0.2	141.77±81.4	**14.76±5.3**	-101.370

Table II

COMPARISON WITH GIST AND OUR DC PROXIMAL NEWTON ON REAL-WORLD BENCHMARK PROBLEMS. THE FIRST COLUMNS OF THE TABLE PROVIDE THE NAME OF THE DATASETS, THEIR STATISTICS. PERFORMANCES REPORTED IN **BOLD** ARE STATISTICALLY SIGNIFICANTLY DIFFERENT THAN THEIR COMPETITOR COUNTERPART ACCORDING TO A WILCOXON SIGNED RANK TEST WITH A P-VALUE AT 0.05. A MINUS SIGN IN THE RELATIVE OBJECTIVE VALUE INDICATES THAT THE DC PROXIMAL NEWTON APPROACH PROVIDES LARGER OBJECTIVE VALUE THAN GIST.

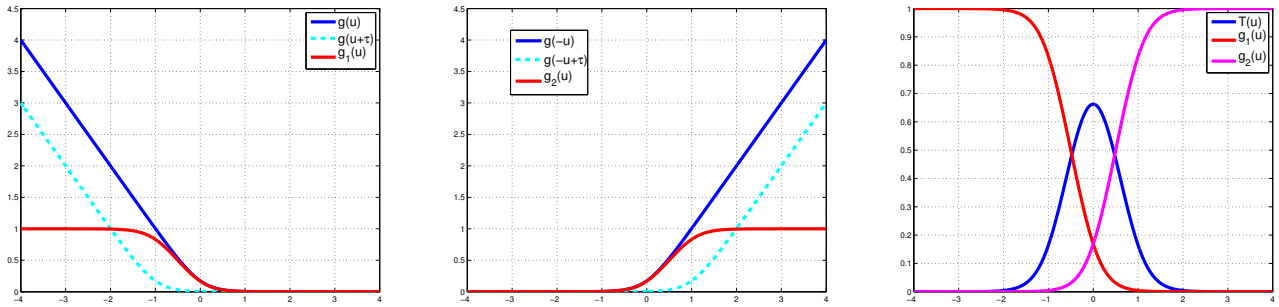

Figure 1. Example of a non-convex smooth transductive loss function $T(\cdot)$ obtained with $\tau = 1$ as well as its components. (left) $g_1(u)$, (middle) $g_2(u)$, (right) DC decomposition of $T(u)$.

provided code was not able to provide a solution in a reasonable amount of time. Again the benefits of unlabeled examples are compelling especially when few labeled examples are in play.

V. CONCLUSIONS

This paper introduced a general proximal Newton algorithm that optimizes the composite sum of functions. A specificity of the approach is its ability to deal with the non-convexity of both terms while one of these terms is in addition allowed to be non-differentiable. While most of the works in the machine learning and optimization communities have been addressing these non-differentiability and non-convexity issues separately, there exists a number of learning problems such as sparse transductive learning that require efficient optimization scheme on non-convex and non-differentiable functions. Our algorithm is based on two steps: the first one looks for a

			Class. Rate (%)	
dataset	ℓ	ℓ_u	Sparse Log	Sparse Transd.
la2	61	2398	67.65±2.6	**70.23±3.1**
sports	85	6778	81.26±5.0	**88.15±4.4**
classic	70	5604	72.74±4.3	**86.97±2.2**
ohscal	55	8873	70.35±2.4	**73.39±3.6**
real-sim	723	57124	88.81±0.3	88.91±1.4

Table III

COMPARING THE RECOGNITION RATE OF A SPARSE LOGISTIC REGRESSION AND A SPARSE TRANSDUCTIVE LOGISTIC REGRESSION BOTH WITH CLIPPED-ℓ_1 REGULARIZER. ℓ AND ℓ_u RESPECTIVELY DENOTES THE NUMBER OF LABELED AND UNLABELED EXAMPLES.

search direction through a proximal Newton step while the second one performs a line search on that direction. We also provide in this work the proof that the iterates generated by this algorithm behaves correctly in the sense that limit points of the sequences are stationary points. Numerical experiments show that the second order information used in our algorithm through the matrix $\mathbf{H}_k$ allow faster convergence than proximal gradient based descent approaches for non-convex regularizers. One of the strength of our framework is its ability to handle non-convexity on both the smooth loss function and the regularizer. We have illustrated this ability by learning a sparse transductive logistic regression model.

For the sake of reproducible research, the code source of the numerical simulation will be freely available on the authors website.

VI. Appendix

A. Lemma 3 and proof

Lemma 4: For $\mathbf{x}$ in the domain of f and assuming that $\mathbf{H}_k \succeq m\mathbf{I}$ with $m > 0$ and ∇f_1 is Lipschitz with constant L then the sufficient condition in Equation (17) holds for all t_k so that

$$t_k \leq \min\left(1, 2m\frac{1-\alpha}{L}\right)$$

Proof : Recall that $\mathbf{x}_+ := \mathbf{x}_k + t_k\Delta\mathbf{x}_k$. By definition, we have

$$\begin{aligned} F(\mathbf{x}_+) - F(\mathbf{x}) = f_1(\mathbf{x}_+) - f_1(\mathbf{x}) - f_2(\mathbf{x}_+) + f_2(\mathbf{x}) \\ + h_1(\mathbf{x}_+) - h_1(\mathbf{x}) - h_2(\mathbf{x}_+) + h_2(\mathbf{x}). \end{aligned}$$

Then by convexity of f_2, h_2 and h_1, we derive that (see equation (14))

$$\begin{aligned} F(\mathbf{x}_+) - F(\mathbf{x}) \leq f_1(\mathbf{x}_+) - f_1(\mathbf{x}) + (1-t)h_1(\mathbf{x}) \\ + th_1(\mathbf{x}+\Delta\mathbf{x}) - (\mathbf{z}_{f_2} + \mathbf{z}_{h_2})^\top(t\Delta\mathbf{x}) \\ - h_1(\mathbf{x}) \end{aligned}$$

According to a Taylor-Laplace formulation, we have :

$$f_1(\mathbf{x}_+) - f_1(\mathbf{x}) = \int_0^1 \nabla f_1(\mathbf{x} + st\Delta\mathbf{x})^\top(t\Delta\mathbf{x})ds$$

thus, we can rewrite

$$\begin{aligned} F(\mathbf{x}_+) - F(\mathbf{x}) &\leq \int_0^1 \nabla f_1(\mathbf{x} + st\Delta\mathbf{x})^\top(t\Delta\mathbf{x})ds - th_1(\mathbf{x}) \\ &\quad + th_1(\mathbf{x}+\Delta\mathbf{x}) - (\mathbf{z}_{f_2} + \mathbf{z}_{h_2})^\top(t\Delta\mathbf{x}) \\ &\leq \int_0^1 \Big(\nabla f_1(\mathbf{x} + st\Delta\mathbf{x}) - \nabla f_1(\mathbf{x})\Big)^\top(t\Delta\mathbf{x})ds \\ &\quad + th_1(\mathbf{x}+\Delta\mathbf{x}) + \nabla f_1(\mathbf{x})^\top(t\Delta\mathbf{x}) \\ &\quad - (\mathbf{z}_{f_2} + \mathbf{z}_{h_2})^\top(t\Delta\mathbf{x}) - th_1(\mathbf{x}) \\ &\leq t\Big(\int_0^1 \Big(\nabla f_1(\mathbf{x} + st\Delta\mathbf{x}) - \nabla f_1(\mathbf{x})\Big)^\top(\Delta\mathbf{x})ds \\ &\quad + h_1(\mathbf{x}+\Delta\mathbf{x}) + \nabla f_1(\mathbf{x})^\top(\Delta\mathbf{x}) \\ &\quad - (\mathbf{z}_{f_2} + \mathbf{z}_{h_2})^\top(\Delta\mathbf{x}) - h_1(\mathbf{x})\Big) \end{aligned}$$

Then using Cauchy-Schwartz inequality and the fact that f_1 is gradient Lipschitz of constant L, we have :

$$\begin{aligned} F(\mathbf{x}_+) - F(\mathbf{x}) &\leq t\Big(\int_0^1 stL\|\Delta\mathbf{x}\|_2^2 ds \\ &\quad + h_1(\mathbf{x}+\Delta\mathbf{x}) + \nabla f_1(\mathbf{x})^\top(\Delta\mathbf{x}) \\ &\quad - (\mathbf{z}_{f_2} + \mathbf{z}_{h_2})^\top(\Delta\mathbf{x}) - h_1(\mathbf{x})\Big) \\ &\leq t\Big(\frac{tL}{2}\|\Delta\mathbf{x}\|_2^2 \\ &\quad + h_1(\mathbf{x}+\Delta\mathbf{x}) + \nabla f_1(\mathbf{x})^\top(\Delta\mathbf{x}) - \\ &\quad (\mathbf{z}_{f_2} + \mathbf{z}_{h_2})^\top(\Delta\mathbf{x}) - h_1(\mathbf{x})\Big) \\ &\leq t\Big(\frac{tL}{2}\|\Delta\mathbf{x}\|_2^2 \\ &\quad + h_1(\mathbf{x}+\Delta\mathbf{x}) - h_1(\mathbf{x}) + \mathbf{v}_k^\top(\Delta\mathbf{x})\Big) \\ &\leq t\Big(\frac{tL}{2}\|\Delta\mathbf{x}\|_2^2 + D\Big) \end{aligned}$$

Now, if t is so that

$$t \leq 2m\frac{1-\alpha}{L}$$

then

$$\begin{aligned} \frac{Lt}{2}\|\Delta\mathbf{x}\|_2^2 &\leq m(1-\alpha)\|\Delta\mathbf{x}\|_2^2 \\ &= (1-\alpha)\Delta\mathbf{x}^\top(m\mathbf{I})\Delta\mathbf{x} \\ &\leq (1-\alpha)\Delta\mathbf{x}^\top\mathbf{H}\Delta\mathbf{x} \\ &\leq -(1-\alpha)D \end{aligned}$$

where the last inequality comes from the descent property. Now, we plug this inequalityback and get

$$t\Big(\frac{tL}{2}\|\Delta\mathbf{x}\|_2^2 + D\Big) \leq t\Big(-(1-\alpha)D + D\Big) = t\alpha D$$

which concludes the proof that for all

$$t \leq \min\left(1, 2m\frac{1-\alpha}{L}\right)$$

we have

$$F(\mathbf{x}_+) - F(\mathbf{x}) \leq t\alpha D$$

References

[1] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society*, vol. 46, pp. 267–288, 1996.

[2] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal Scientific Comput.*, vol. 20, no. 1, pp. 33–61, 1999.

[3] Y. Li and S.-I. Amari, "Two conditions for equivalence of 0-norm solution and 1-norm solution in sparse representation," *Neural Networks, IEEE Transactions on*, vol. 21, no. 7, pp. 1189–1196, July 2010.

[4] D. Donoho, "For most large underdetermined systems of linear equations, the minimal ℓ_1 solution is also the sparsest solution," *Communication in Pure and Applied Mathematics*, vol. 59, no. 6, pp. 797–829, 2006.

[5] S. Shevade and S. Keerthi, "A simple and efficient algorithm for gene selection using sparse logistic regression," *Bioinformatics*, vol. 19, no. 17, pp. 2246–2253, 2003.

[6] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, pp. 183–202, 2009.

[7] G.-X. Yuan, C.-H. Ho, and C.-J. Lin, "An improved glmnet for l1-regularized logistic regression," *The Journal of Machine Learning Research*, vol. 98888, pp. 1999–2030, 2012.

[8] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Convex optimization with sparsity-inducing norms," *Optimization for Machine Learning*, 2011.

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

[10] 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.

[11] K. Knight and W. Fu, "Asymptotics for lasso-type estimators," *Annals of Statistics*, vol. 28, pp. 1356–1378, 2000.

[12] E. Candès, M. Wakin, and S. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. Fourier Analysis and Applications*, vol. 14, pp. 877–905, 2008.

[13] L. Laporte, R. Flamary, S. Canu, S. Dejean, and J. Mothe, "Nonconvex regularizations for feature selection in ranking with sparse svm," *Neural Networks and Learning Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2013.

[14] G. Gasso, A. Rakotomamonjy, and S. Canu, "Recovering sparse signals with a certain family of non-convex penalties and dc programming," *IEEE Trans. Signal Processing*, vol. 57, no. 12, pp. 4686–4698, 2009.

[15] P. Combettes and J. Pesquet, "Proximal splitting methods in signal processing," *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pp. 185–212, 2011.

[16] J. Lee, Y. Sun, and M. Saunders, "Proximal newton-type methods for convex optimization," in *Advances in Neural Information Processing Systems*, 2012, pp. 836–844.

[17] S. Becker and J. Fadili, "A quasi-newton proximal splitting method," in *Advances in Neural Information Processing Systems*, 2012, pp. 1–9.

[18] H. A. Le Thi and T. Pham Dinh, "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, pp. 23–46, 2005.

[19] T. Pham Dinh and H. A. Le Thi, "Convex analysis approach to dc programming: Theory, algorithms and applications," *Acta Mathematica Vietnamica*, vol. 22, no. 1, pp. 287–355, 1997.

[20] F. Akoa, "Combining dc algorithms (dcas) and decomposition techniques for the training of nonpositive semidefinite kernels," *Neural Networks, IEEE Transactions on*, vol. 19, no. 11, pp. 1854–1872, Nov 2008.

[21] M. Figueiredo, R. Nowak, and S. Wright, "Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics in Signal Processing: Special Issue on Convex Optimization Methods for Signal Processing*, vol. 1, no. 4, pp. 586–598, 2007.

[22] G. Golub and C. Van Loan, *Matrix computations*. Johns Hopkins University Press, 1996, vol. 3.

[23] P.-L. Loh and M. J. Wainwright, "Regularized m-estimators with nonconvexity: Statistical and algorithmic theory for local optima," in *Advances in Neural Information Processing Systems 26*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds., 2013, pp. 476–484.

[24] T. Pham Dinh and H. A. Le Thi, "Dc optimization algorithms for solving the trust region subproblem," *SIAM Journal of Optimization*, vol. 8, pp. 476–505, 1998.

[25] H. Mine and M. Fukushima, "A minimization method for the sum of a convex function and a continuously differentiable function," *Journal of Optimization Theory and Applications*, vol. 33, no. 1, pp. 9–23, 1981.

[26] P. Gong, C. Zhang, Z. Lu, J. Huang, and Y. Jieping, "A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013, pp. 37–45.

[27] Z. Lu, "Sequential convex programming methods for a class of structured nonlinear programming," ArXiv:1210.3039, Tech. Rep., 2012.

[28] E. Chouzenoux, J.-C. Pesquet, and A. Repetti, "Variable metric forward-backward algorithm for minimizing the sum of a differentiable function and a convex function," Institut Gaspard Monge, Tech. Rep., 2013.

[29] S. Sra, "nonconvex proximal splitting : batch and incremental algorithms," in *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[30] T. Zhang, "Analysis of multi-stage convex relaxation for sparse regularization," *Journal of Machine Learning Researc*, vol. 11, pp. 1081–1107, 2010.

[31] A. Boisbunon, R. Flamary, and A. Rakotomamonjy, "Active set strategy for high-dimensional non-convex sparse optimization problems," in *International Conference on Acoustic, Speech and Signal Processing*, 2014.

[32] A. Rakotomamonjy, R. Flamary, G. Gasso, and S. Canu, "$\ell_p - \ell_q$ penalty for sparse linear and sparse multiple kernel multi-task learning,," *IEEE Trans. on Neural Networks*, vol. 22, no. 8, pp. 1307–1320, 2011.

[33] T. Joachims, "Transductive inference for text classification using svms," in *Proceedings of The 16th International Conference on Machine Learning*, 1999.

[34] R. Collobert, F. Sinz, J. Weston, and L. Bottou, "Large scale transductive svms," *Journal of Machine Learning Research*, vol. 7, pp. 1687–1712, 2006.

[35] O. Chapelle and A. Zien, "Semi-supervised classification by low density separation," in *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistic*, 2005, pp. 57–64.