On Non-differentiable Time-varying Optimization

Andrea Simonetto and Geert Leus
Faculty of EEMCS, Delft University of Technology, 2826 CD Delft, The Netherlands
e-mail: {a.simonetto, g.j.t.leus}@tudelft.nl

Abstract—We consider non-differentiable convex optimization problems that vary continuously in time and we propose algorithms that sample these problems at specific time instances and generate a sequence of converging near-optimal decision variables. This sequence converges up to a bounded error to the solution trajectory of the time-varying non-differentiable problems. We illustrate through analytical examples and a realistic numerical simulation the benefit of the algorithms in signal processing applications, e.g., for reconstructing time-varying sparse signals.

I. INTRODUCTION

Non-differentiable convex optimization is ubiquitous in many signal processing, communication and networking problems; take for example compressed sensing [1], which has been extensively studied recently. The standard convex programming approaches to solve such problems work quite well provided that the optimization program to be solved is stationary, that is: fixed at design time. When the optimization program changes in time, e.g., the sparse signal to be reconstructed is changing, then standard approaches may require too much computational resources to solve the problem at a specific time instance, and they might fail to keep up with the time changes.

In this paper, we specifically consider time-varying non-differentiable convex optimization problems of the form

\[
\min_{x \in \mathbb{R}^n} f(x; t), \quad \forall t \geq 0. 
\]  

where \( x \) is the vector of decision variables, while the function \( f : \mathbb{R}^n \rightarrow \mathbb{R}_+ \) is a non-differentiable, yet strongly convex, cost function parametrized by the time \( t \). Our aim is to find the optimal decision vector at each time instance, i.e., the solution trajectory \( x^*(t) \). Given the nature of the problem at hand, one could think of sampling it at different time instances \( t_k \), with \( k = 0, 1, 2, \ldots, \) and then solving the time-invariant problems

\[
\min_{x \in \mathbb{R}^n} f(x; t_k), 
\]

for all \( k \)'s. Even for moderate problem sizes, this approach might be not computationally viable (since solving each instance of the problem at high accuracy might require more time than the sampling time). It makes sense therefore to consider approximate, or running, schemes, where each instance of the problem (i.e., for a fixed \( k \)) is solved only approximatively. Research in this direction can be found in [2]–[5] (and references therein), but is all limited to differentiable cost functions. In many cases, these works even require second, third, and (sometimes) higher order derivatives to exist and be sufficiently “well-behaved”.

In this paper, we consider non-differentiable problems in their generality and we study running methods to find and track the solution trajectory \( x^*(t) \) up to a bounded error, depending on the variability of the optimizers in time. In particular, we leverage monotone operators [6], contractions [7], and the alternating directions method of multipliers (ADMM) [8], [9].

We propose two algorithms, one for non-differentiable strongly convex functions with bounded subgradients, based on subgradient descent, and one for strong convex functions only, relying on generalized ADMM. We also work out some interesting examples where our algorithms can be applied, namely \( \ell_1 \)-regularized problems and constrained problems. Finally, we display the benefit of our methods in a realistic numerical setting coming from sparse signal reconstruction.

Notation. For any vector \( x \in \mathbb{R}^n \), we indicate as \( ||x|| \) the Euclidean norm. Let the function \( f : C \subseteq \mathbb{R}^n \rightarrow \mathbb{R} \) be convex, closed, and proper (CCP). The subdifferential mapping of \( f \) is indicated as \( \partial f \) and it is defined by

\[
\partial f := \{ (x, h) | x \in C, \forall y \in C, f(y) \geq f(x) + h^T(y-x) \}. 
\]

The set \( \partial f(x) \) is the subdifferential set of \( f \) at \( x \in C \). Any vector \( h \in \partial f(x) \) is called a subgradient of \( f \) at \( x \). The proximal mapping of the function \( f \) is

\[
\text{prox}_{f,\beta}(x) := \arg \min_y \{ f(y) + \beta/2 ||y-x||^2 \}. 
\]

The soft-thresholding mapping is

\[
S_\beta(x) := (x-\beta)_+ - (x-\beta)_+, \quad \text{with} \ (\cdot)_+ = \max\{\cdot, 0\}. 
\]

II. PROBLEM FORMULATION AND CORE IDEAS

We aim at finding and tracking the solution trajectory \( x^*(t) \) of the problem (1) by sampling it at specific time instances \( t_k \), with \( k = 0, 1, 2, \ldots \). In particular, we construct algorithms that generate a sequence of near-optimal decision variables \( \{x_k\} \) that are required to eventually converge as

\[
\lim_{k \rightarrow \infty} ||x_k - x^*(t_k)|| \leq \epsilon, 
\]

where \( \epsilon \) is the asymptotical error bound.

The core ideas that we use in this paper are two: the first is the construction of algorithms that describe contraction in their time-invariant version, i.e., algorithms for which we know that

\[
||x_{k+1} - x^*(t_k)|| \leq \rho ||x_k - x^*(t_k)|| 
\]

for a positive \( \rho < 1 \). This yields the convergence properties of the algorithms. The second is to assume that the distance between optimizers at subsequent time instances is upper bounded.

Assumption 1: There exists a constant \( \sigma_x \), for which

\[
||x^*(t_{k+1}) - x^*(t_k)|| \leq \sigma_x, \quad \forall k \geq 0. 
\]

This assumption is responsible for the asymptotical error.

III. STRONGLY CONVEX AND BOUNDED SUBGRADIENT

We start our analysis with the assumption that the convex function \( f(x; t) \) is strongly convex for all \( x \in \mathbb{R}^n \) and uniformly in \( t \), and that the subgradient of \( f(x; t) \) is bounded.

Assumption 2: (Strong convexity and boundedness)
(i) The function \( f(x; t) \) is \( m \)-strongly convex for all \( x \) and uniformly in \( t \), that is: the function \( f(x; t) - m\|x\|^2 \) is convex for all \( x \) and uniformly in \( t \).

(ii) The subgradient \( \partial f(x; t) \) is bounded for all \( x \) and uniformly in \( t \), that is: there exists a finite constant \( C \) for which \( \|\partial f(x; t)\| \leq C \) for all \( x \) and uniformly in \( t \).

We notice that Assumption 2 does not require differentiability of the cost function \( f(x; t) \). With this assumption in place, we know that the subgradient method describes an approximate contraction with respect to the Euclidean norm \( \| \cdot \| \) up to a bounded error term, [10, Proposition 2.4]. We use this fact to construct our first algorithm (Algorithm 1), in which \( \partial f_k \) is the subdifferential of the function \( f \) at time \( t_k \). Its convergence goes as follows.

**Theorem 1:** Let the Assumptions 1 and 2 hold. Let \( \{x_k\} \) be the sequence generated by Algorithm 1 with stepsize \( \alpha \) satisfying

\[
\epsilon/(2m(\epsilon + 1)) < \alpha < 1/2m,
\]

for any choice of \( \epsilon > 0 \). Let the coefficient \( \rho \) be defined as \( \rho = (1 - 2m\alpha)(1 + \epsilon) < 1 \). Then, convergence of \( \{x_k\} \) to the solution trajectory \( x^*(t_k) \) is Q-linear as

\[
\|x_k - x^*(t_k)\|^2 \leq \rho^k\|x_0 - x^*(t_0)\|^2 + \left[\alpha^2C^2(1 + \epsilon) + \sigma_\varepsilon^2(1 + 1/\epsilon)/(1 - \rho^k)/(1 - \rho),
\]

while the asymptotical error floor is upper bounded as

\[
\lim_{k \to \infty} \|x_k - x^*(t_k)\| \leq \sqrt{\alpha^2C^2(1 + \epsilon) + \sigma_\varepsilon^2(1 + 1/\epsilon)/(1 - \rho)}.
\]

**Proof:** (Sketch) The optimality gap at time \( k \) is

\[
\|x_{k+1} - x^*(t_{k+1})\|^2 \leq (1 + \epsilon)\|x_{k+1} - x^*(t_k)\|^2 + (1 + 1/\epsilon)\|x^*(t_{k+1}) - x^*(t_k)\|^2
\]

\[
\leq (1 + \epsilon)\|x_{k+1} - x^*(t_k)\|^2 + (1 + 1/\epsilon)\sigma_\varepsilon^2,
\]

where we have used Peter-Paul inequality, which is valid for any \( \epsilon > 0 \), and Assumption 1. We use now the contraction property of the subgradient method [10, Proposition 2.4].

\[
\|x_{k+1} - x^*(t_k)\|^2 \leq (1 - 2m\alpha)\|x_k - x^*(t_k)\|^2 + \alpha^2C^2.
\]

By using (12) in (11), under condition (8), the claims (9) and (10) follow.

Theorem 1 ensures that Algorithm 1 generates a converging sequence of near optimal decision variables \( \{x_k\} \), despite the fact that the cost function is changing over time. Given the strong convexity of the cost function, convergence to the optimal trajectory \( x^*(t_k) \) is Q-linear and up to a bound depending on the variability of the optimizer in time and the stepsize.

Theorem 1 relies on a bound of the subgradient, which also yields an additional asymptotic error. In addition, the stepsize condition (8) might be restrictive. In the next section, we see how to weaken our requirements.

**IV. SUM OF CONVEX FUNCTIONS**

We now focus on time-varying problems of the form

\[
\min_{x \in \mathbb{R}^n} f(x; t) := \varphi(x; t) + \psi(Ax; t), \quad \forall t \geq 0
\]

where both \( \varphi \) and \( \psi \) are time-varying CCP functions, and where the matrix \( A \in \mathbb{R}^{m \times n} \). The underlying idea of analyzing these class of problems is to weaken the subgradient assumption (Assumption 2) by splitting the cost into a sum of two functions holding different properties. In particular, we substitute Assumption 2 with the following.

**Assumption 3:** (Weak smoothness) The function \( \varphi(x; t) \) is differentiable, \( m \)-strongly convex and with \( L \)-Lipschitz continuous gradient for all \( x \) and uniformly in \( t \).

As we can see, no assumption is made on \( \psi \), besides being a CCP function, therefore the resulting time-varying cost function \( f \) is still a non-differentiable function, which by Assumption 3 is \( m \)-strongly convex in \( x \) and uniformly in \( t \), but its subgradient is not bounded, in general. To handle this class of problems, we will use a generalized form of ADMM, to leverage its linear convergence (i.e., contraction) properties.

First, we define the auxiliary variable \( y = Ax \), and then rewrite the time-varying optimization problem (13) in the equivalent (lifted) form

\[
\min_{x \in \mathbb{R}^n, y \in \mathbb{R}^m} F(x, y; t) := \varphi(x; t) + \psi(y; t), \quad \forall t \geq 0
\]

Second, we can derive the augmented Lagrangian function associated with the lifted problem (14) as

\[
\mathcal{L}(x, y, \lambda; t) = F(x, y; t) + \lambda^T(y - Ax) + \beta/2\|y - Ax\|^2,
\]

where the vector \( \lambda \in \mathbb{R}^m \) is the dual variable associated with the equality constraint \( y = Ax \), and where the positive scalar \( \beta \) weights the augmented quadratic term.

Third, we can sample the Lagrangian at different time instances \( t_k \), we can introduce sequences of primal and dual variables as \( \{x_k, y_k, \lambda_k\} \), and solve the time-varying problem (13) with the following recursion

\[
x_{k+1} = \min_{x \in \mathbb{R}^n} \mathcal{L}(x, y_k, \lambda_k; t_k) + 1/2\|x - x_k\|^2
\]

\[
y_{k+1} = \min_{y \in \mathbb{R}^m} \mathcal{L}(x_{k+1}, y, \lambda_k; t_k) + 1/2\|y - y_k\|^2
\]

\[
\lambda_{k+1} = \lambda_k + \beta(y_{k+1} - Ax_{k+1}),
\]

for a choice of positive definite matrices \( Q \) and \( P \). This recursion is a generalization of ADMM, where the matrices \( Q \) and \( P \) have the task of weighing the past iterates and of inducing better convergence properties. For an account of this type of generalization of ADMM in the case of time-invariant cost functions, we refer to [11]. The underlying reason of using such a recursion as building block of our time-varying algorithm is (once again) its contraction properties: define the stacked vector \( u_k = [x_k^T, y_k^T, \lambda_k^T]^T \), and its optimal value at \( t_k \), i.e., \( u^*(t_k) \), then (when \( A \) is full row rank) we can show that

\[
\|u_{k+1} - u^*(t_k)\| \leq \rho\|u_k - u^*(t_k)\|,
\]

for a certain \( \rho < 1 \), see [11, Scenario 1, Case 4]. With this in place, we can derive Algorithm 2, which enjoys the following convergence properties.

**Theorem 2:** Let Assumptions 1 and 3 hold. Let \( \{x_k, y_k, \lambda_k\} \) be the sequence generated by Algorithm 2 for a particular choice.
of $\beta > 0$, $P > 0$, and $Q > 0$. Define the stacked vector $u_k = (x_k^T, y_k^T, \lambda_k^T)^T$, and its optimal value at $t_k$ as $u^*(t_k)$. In addition, assume that

(i) The matrix $A$ is full row rank.

(ii) The difference of the dual variable at two subsequent time instances is upper bounded, i.e., $|\Lambda^*(t_{k+1}) - \Lambda^*(t_k)| \leq \sigma \lambda_k$.

Then the sequence $\{u_k\}$ converges Q-linearly to $u^*(t_k)$ as

$$\|u_{k+1} - u^*(t_{k+1})\| \leq \rho^k \|u_0 - u^*(t_0)\| + \tilde{\sigma}(1 - \rho^k)/(1 - \rho),$$

for a certain $\rho < 1$ and with $\tilde{\sigma} = \sqrt{\sigma_x^2 + \|A\|^2 \sigma_y^2 + \sigma_{\lambda}^2}$, while the asymptotical error floor is upper bounded as

$$\lim_{k \to \infty} \|u_{k+1} - u^*(t_{k+1})\| \leq \tilde{\sigma}/(1 - \rho).$$  \hspace{1cm} (19)

**Proof:** (Sketch) The proof follows the same arguments of the proof of Theorem 1. In particular, one can use the contraction property (17) [cfr. [11]] and the Triangle inequality to say

$$\|u_{k+1} - u^*(t_{k+1})\| \leq \rho \|u_k - u^*(t_k)\| + \tilde{\sigma},$$

where $\tilde{\sigma}$ encodes the bound on the variability of the primal and dual variables in time and the contraction parameter $\rho < 1$ is given in [11]. With this in place, simple algebra yields the claims (18) and (19).

We notice that the algorithm RgADMM involves two optimization problems per time step $k$. Those problems are however easier to be solved than the original one (13); in many cases, they can be solved in closed form, or they are highly parallelizable. Thus the computational effort per time step $k$ stays limited [8].

V. EXAMPLES

A. Time-varying optimization with $\ell_1$ regularization

The first example we focus on is time-varying optimization with $\ell_1$ regularization. It is hard to over stress how important and widespread this particular problem has become recently (in its time-invariant version), particularly in signal processing. In this section, we consider the following variant

$$\min_{x \in \mathbb{R}^n} f(x; t) = \varphi(x; t) + \mu(t) \|x\|_1, \quad \forall t \geq 0,$$  \hspace{1cm} (21)

where $\mu(t)$ is a time-varying positive scalar, and we work under the weak smoothness assumption (Assumption 3). We proceed as done in Section IV: we derive a lifted problem with the equality constraint $y = x$ and calculate the augmented Lagrangian as

$$\mathcal{L}(x, y, \lambda; t) = \varphi(x; t) + \mu(t) \|y\|_1 + \lambda^T(x - y) + \frac{\beta}{2} \|x - y\|^2.$$  \hspace{1cm} (22)

We further define the sampled time-varying function $g_k(x)$ as,

$$g_k(x) = \varphi(x; t_k) + \lambda_k^T x + \frac{\beta}{2} \|x - x_k\|^2.$$  \hspace{1cm} (23)

with $q > 0$. By using the proposed RgADMM algorithm with $P = pI$ and $Q = qI$, we arrive at the following recursion

$$x_{k+1} = \text{prox}_{g_k, \beta}(x)$$  \hspace{1cm} (24a)

$$y_{k+1} = \frac{1}{\beta + p} \sum_{i=1}^q S_{\mu(t_k)}(\lambda_k + \beta x_{k+1} + py_k(i)), \ i \in \{1, n\}$$  \hspace{1cm} (24b)

$$\lambda_{k+1} = \lambda_k + \beta(y_{k+1} - x_{k+1})$$  \hspace{1cm} (24c)

for which $\text{prox}_{g, \beta}$ and $S_{\mu(t)}$ are defined in (3)-(4), and the notation $x_k(i)$ represents the $i$-th component of the vector $a_k$.

B. Time-varying differentiable constrained optimization

Another interesting example of the optimization problem (13) is the case of differentiable constrained optimization, as

$$\min_{x \in \mathbb{C}(t)} \varphi(x; t), \quad \forall t \geq 0,$$  \hspace{1cm} (25)

where the time-varying function $\varphi(x; t)$ is differentiable and strongly convex, while the set $\mathbb{C}(t)$ is a closed, non-empty, convex set. One can see that, by introducing the indicator function $\iota_{\mathbb{C}(t)}(x)$ which is defined as

$$\iota_{\mathbb{C}(t)}(x) := 0 \quad \text{if} \quad x \in \mathbb{C}(t) \quad \text{and} \quad \iota_{\mathbb{C}(t)}(x) := \infty, \quad \text{otherwise},$$

then (25) can be written as the unconstrained program

$$\min_{x \in \mathbb{R}^n} f(x; t) = \varphi(x; t) + \iota_{\mathbb{C}(t)}(x), \quad \forall t \geq 0,$$  \hspace{1cm} (26)

which is in the form of (13). By lifting this problem with the additional variable $y = x$, and by using Algorithm 2 with $P = pI$, $Q = qI$, we can derive the time-varying recursion

$$x_{k+1} = \text{prox}_{g_k, \beta}(x)$$  \hspace{1cm} (27a)

$$y_{k+1} = \arg\min_{y \in \mathbb{C}(t)} \{p\|y - y_k\|^2 + \beta\|y - x_{k+1}\|^2 - 2\lambda_k^Ty\}$$  \hspace{1cm} (27b)

$$\lambda_{k+1} = \lambda_k + \beta(y_{k+1} - x_{k+1})$$  \hspace{1cm} (27c)

The algorithm defined by (27) solves the time-varying problem (25) up to a bounded asymptotical error.

VI. NUMERICAL EXAMPLE

We now focus on a numerical example to showcase the behavior of Algorithm 2 with simulated data. We use a quite realistic example coming from the recent interest in online convex optimization [13], online ADMM [14]–[16], and dynamic compressive sensing [17].

Suppose $x(t) \in \mathbb{R}^n$ is a deterministic time-varying sparse signal to be estimated based on the measurements $z_i(t_k) \in \mathbb{R}$, connected with each other via the linear regression model

$$z_i(t_k) = c_i(t_k)^T x^*(t_k) + v_i(t_k), \quad i = 1, \ldots, m,$$  \hspace{1cm} (28)

where $m$ is the total number of measurements at any time, and $v_i$ is a zero-mean Gaussian noise term. We assume that the regressors $c_i(t_k)$ are known random variables uncorrelated among each other and uncorrelated with $v_i(t_k)$. The estimation problem can be formulated as the following time-varying optimization problem for all $k \geq 1$

$$\min_{x \in \mathbb{R}^n} \left[ \sum_{\tau=1}^k \frac{1}{k} \|C(t_\tau)x - z(t_\tau)\|^2 \right] + \gamma\|x\|^2 + \mu\|x\|_1$$  \hspace{1cm} (29)

where the matrix $C(\tau) \in \mathbb{R}^{m \times n}$ collects the regressor at time $\tau$, i.e., $C(\tau) = (c_1(\tau), \ldots, c_m(\tau))^T$, while the vector $z(\tau) \in \mathbb{R}^m$ collects the measurements, i.e., $z(\tau) = (z_1(\tau), \ldots, z_m(\tau))^T$. In addition, the term $\eta \in (0, 1]$ is a forgetting factor that attempts at approximating a sliding-window expectation. Finally, the regularization term with $\mu > 0$ ($\gamma > 0$) enhances the (group) sparsity of the reconstruction. More details on this estimation problem are given in [12], [18].

Problems like (29) arise in online or adaptive estimation. The case in which $\eta = 1$, $\gamma = \mu = 0$ can be seen as recursive
least-squares. The case $\gamma = 0, \mu > 0$ represents a recursive LASSO formulation [18], while the case that we consider here ($\gamma > 0, \mu > 0$) can be seen as recursive time-varying elastic net [19]. The reason we consider only the case $\gamma > 0$ is that in this way we can form a $\varphi$ function that is strongly convex and verifies Assumption 3. Problem (29) is a discretized instance of the time-varying problem (13) and therefore we can use the RgADMM algorithm to track its solution trajectory. In the numerical example, we use the following values for the simulation parameters: the dimension of the signal is $n = 100$, while at each sampling time we collect $m = 5$ measurements. The number of nonzero elements of $x^*(t)$ is 10 and they evolve according to the continuous dynamics

$$x^{(i)}(t) = \cos(\omega t + 2\pi i/n), \quad \omega = 0.001.$$  \hspace{1cm} (30)$$

The regressors $c_i(t)$ are drawn from a zero-mean Gaussian distribution with covariance 1, while the noise terms $v_i(t)$ are drawn from a zero-mean Gaussian distribution with standard deviation 0.01. The parameters of the RgADMM algorithm are $\beta = 1, \rho = q = \gamma = \mu = 0.1$, while the forgetting factor $\eta$ is set as $\eta = 0.225/h$, being $h = t_k - t_{k-1}$.

In Figure 1, we display the evolution of the sparse signal and its reconstruction via the RgADMM algorithm based on the optimization (29), at different time instances $t_k$ and for a sampling interval of $h = 2$. As we can see, the RgADMM algorithm successfully identifies the nonzero elements and tracks the signal up to a bounded error.

Figure 2 depicts the error in terms of $\|x_k - x^*(t_k)\|$ for different sampling intervals $h$. We see that the error arrives quickly to an error floor which increases with increasing $h$.

In both the figures, we have also depicted the behavior of the algorithm of [12] adapted to solving problem (29). This algorithm has been proven to converge only in time-invariant situations, but, in this time-varying case, it has comparable heuristic performance with our proven RgADMM (only, it seems more affected by noise). A thorough characterization and comparison is left as future research.

REFERENCES


