# OpReg-Boost: Learning to Accelerate Online Algorithms with Operator Regression

**Nicola Bastianello** NICOLA.BASTIANELLO@DEI.UNIPD.IT
*Department of Information Engineering (DEI), University of Padova, Italy*

**Andrea Simonetto** ANDREA.SIMONETTO@ENSTA-PARIS.FR
*UMA, ENSTA Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France*

**Emiliano Dall'Anese** EMILIANO.DALLANESE@COLORADO.EDU
*Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, USA*

**Editors:** R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

## Abstract

This paper presents a new regularization approach – termed *OpReg-Boost* – to boost the convergence of online optimization and learning algorithms. In particular, the paper considers online algorithms for optimization problems with a time-varying (weakly) convex composite cost. For a given online algorithm, OpReg-Boost learns the closest algorithmic map that yields linear convergence; to this end, the learning procedure hinges on the concept of *operator regression*. We show how to formalize the operator regression problem and propose a computationally-efficient Peaceman-Rachford solver that exploits a closed-form solution of simple quadratically-constrained quadratic programs (QCQPs). Simulation results showcase the superior properties of OpReg-Boost w.r.t. the more classical forward-backward algorithm, FISTA, and Anderson acceleration.

**Keywords:** online optimization, operator regression, acceleration, weakly convex

## 1. Introduction

In recent years, the increasing volume of streaming data in many engineering and science domains has stimulated a growing number of research efforts on online optimization and learning (Popkov (2005); Besbes et al. (2015); Asif and Romberg (2014); Hall and Willett (2015); Jadbabaie et al. (2015); Mokhtari et al. (2016); Dall'Anese et al. (2020); Li et al. (2020) and many others). In data processing and machine learning applications, the cost function and the constraints (if present) are parametrized over data points that arrive sequentially; consequently, cost and constraint are time-dependent to reflect new data points and possibly time-varying learning objectives. Beyond data processing and machine learning applications, emerging problems in the context of learning-based control have stimulated lines of research in online identification of dynamical systems Zheng and Li (2021), and online optimization for robotics Berkenkamp et al. (2016); Luo et al. (2020), model predictive control Paternain et al. (2018); Liao-McPherson et al. (2018); Zhang et al. (2021), and games Belgioioso et al. (2021); Fabiani et al. (2021), to name a few.

Let now $k \in \mathbb{N}$ and $F_k(\boldsymbol{x})$ be a time-varying function, then formally we are interested in time-varying problems of the form

$$\boldsymbol{x}_k^* \in \underset{\boldsymbol{x} \in \mathbb{R}^n}{\arg\min}\, F_k(\boldsymbol{x}) := f_k(\boldsymbol{x}) + g_k(\boldsymbol{x}) \tag{1}$$

In particular, we assume that $f_k : \mathbb{R}^n \to \mathbb{R}$ is closed, proper, and $\mu$-weakly convex[1] for each $k \in \mathbb{N}$, and $g_k : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is closed, convex and proper uniformly in time (optionally, one can also consider a setting where $g_k \equiv 0$). The goal is to design an *online algorithm* $\mathcal{A}_k : \mathbb{R}^n \to \mathbb{R}^n$, with updates $\boldsymbol{x}_k = \mathcal{A}_k(\boldsymbol{x}_{k-1})$, so that the sequence $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ exhibits an asymptotic behavior $\limsup_{k\to\infty} F_k(\boldsymbol{x}_k) - F_k^* \leq B < \infty$, for a properly defined sequence of optimal value functions $\{F_k^*\}_{k\in\mathbb{N}}$ and with $B$ as small as possible. For this result to be feasible, a blanket assumption common in the online optimization literature is that the variations of problem in time (in terms of path length or functional variability) can be upper bounded by a sub-linear or a linear function of $k$; see *e.g.*, Besbes et al. (2015); Jadbabaie et al. (2015); Dall'Anese et al. (2020); Mokhtari et al. (2016); Li et al. (2020); Hallak et al. (2020) . If this latter function is linear in $k$, then it is known that online algorithms exhibit an asymptotic error.

A key intuition is to use the existence of this error as an advantage: given the presence of an error due to the dynamics of the cost, one can leverage regularizations in the optimization problem or modifications of the algorithmic steps to boost the convergence without necessarily sacrificing performance. Surprisingly, *there may be no trade-off between accuracy and convergence*; for example, algorithms constructed based on the regularized problems may offer superior convergence and lower asymptotical errors w.r.t. algorithms built based on the original problem, even though the set of optimal solutions is explicitly perturbed. This line of thought stemmed in the static domain from the seminal works Nesterov (2005); Koshal et al. (2011); Devolder et al. (2012), and more recently in the online setting Simonetto and Leus (2014); Bastianello et al. (2020).

By building on this, a natural question is "*how to best design a surrogate algorithm that allows a gain in convergence rate without compromising optimality?*".

To answer this question, one possibility is to modify the cost function by substituting it with a surrogate function that is, for example, strongly convex and smooth. To fix the idea, consider a non-convex function $f : \mathbb{R} \to \mathbb{R}$ as in Figure 1. One can evaluate the function at specific points (grey dots) and fit the functional evaluations with a strongly convex function $\hat{f}$. As long as $f$ and $\hat{f}$ are not "dramatically different", the reasoning is that solving the problem of minimizing $\hat{f}$ instead of $f$ will then give the algorithm a boost in terms of convergence rate (without leading to a larger asymptotical error). For this option, which we term Convex Regression, see Bastianello et al. (2021).

In this paper, we focus on a different approach that consists in modifying the algorithmic map $\mathcal{A}_k$. The idea is to substitute $\mathcal{A}_k$ with a surrogate mapping that is the "closest" to $\mathcal{A}_k$ (in a well defined sense) and has given desirable properties; for example, it is a contractive map. In Figure 1, as an example we consider the case of a gradient descent algorithm in terms of a fixed point operator $\mathcal{A}_k = \mathcal{T}_k = \mathcal{I} - \alpha\nabla_{\boldsymbol{x}} f_k$, with $\alpha > 0$ being the step size. The idea here is to use evaluations of $\mathcal{T}_k$ to fit a mapping $\hat{\mathcal{T}}_k$ with useful properties (*e.g.*, contractivity). By using $\hat{\mathcal{T}}_k$ in lieu of $\mathcal{T}_k$, then one may be able to boost convergence and possibly reduce the asymptotical error. We show in our

[1] **Notation**. We say that a function $f : \mathbb{R}^n \to \mathbb{R}$ is $\mu$-weakly convex if $f(x) + \mu/2\|\boldsymbol{x} - \boldsymbol{x}_0\|_2^2$, with $\mu > 0$, is convex. The set of convex functions on $\mathbb{R}^n$ that are $L$-smooth (*i.e.*, have $L$-Lipschitz continuous gradient) and $\mu$-strongly convex is denoted as $\mathcal{S}_{\mu,L}(\mathbb{R}^n)$, for $\mu, L > 0$; $\mathcal{S}_{0,L}(\mathbb{R}^n)$ is the set of $L$-smooth convex functions. An operator $\mathcal{T} : \mathbb{R}^n \to \mathbb{R}^n$ is non-expansive iff $\|\mathcal{T}(\boldsymbol{x}) - \mathcal{T}(\boldsymbol{y})\| \leq \|\boldsymbol{x} - \boldsymbol{y}\|$, for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$; on the other hand, $\mathcal{T} : \mathbb{R}^n \to \mathbb{R}^n$ is $\zeta$-contractive, with $\zeta \in (0, 1)$, iff $\|\mathcal{T}(\boldsymbol{x}) - \mathcal{T}(\boldsymbol{y})\| \leq \zeta\|\boldsymbol{x} - \boldsymbol{y}\|$, for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. We denote the composition of two operators $\mathcal{T}_1, \mathcal{T}_2$ as $(\mathcal{T}_1 \circ \mathcal{T}_2)(\boldsymbol{x}) = \mathcal{T}_1(\mathcal{T}_2(\boldsymbol{x}))$. We denote by $\mathcal{I}$ the identity map $\mathcal{I}(\boldsymbol{x}) = \boldsymbol{x}$. We define as $\mathrm{prox}_{\alpha g}(\boldsymbol{y}) = \arg\min_{\boldsymbol{x}} \left\{ g(\boldsymbol{x}) + \|\boldsymbol{x} - \boldsymbol{y}\|^2 / (2\alpha) \right\}$ the proximal operator of a function $g$ with parameter $\alpha > 0$, and we denote by $\mathrm{proj}_C(\cdot)$, the projection operator onto the set $C$. We denote by $\boldsymbol{I}_n$ the identity matrix of size $n$, and by $\otimes$ the Kronecker product.

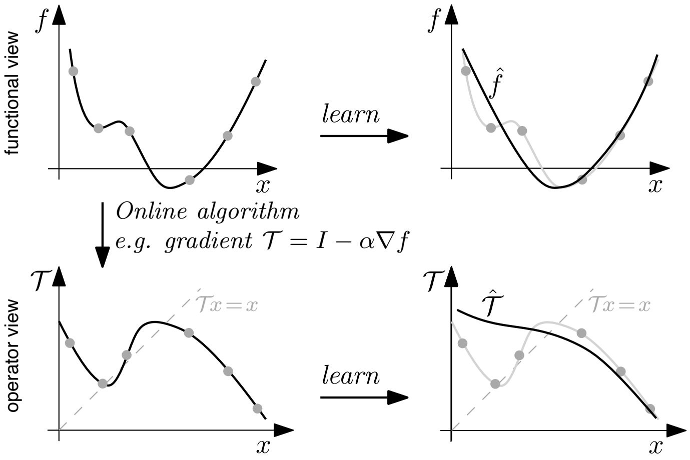

Figure 1: The idea of boosting via projection onto the space of "good" functions or "good" fixed point operators. One can interpret the evaluation of function $f$ or operator $\mathcal{T}$ as noisy evaluations of an underlying "better" function or operator, $\hat{f}$ and $\hat{\mathcal{T}}$, respectively, and use the latter to solve the problem instead. This gives rise to convex-regression-based boosting or operation-regression-based boosting (OpReg-Boost).

numerical experiments that this methodology – referred to as *OpReg-Boost* – outperforms the first option where one utilizes a surrogate cost. Overall, this paper offers the following contributions.

1. We present a novel OpReg-Boost method to *learn-project-and-solve* with linear convergence optimization problems. The method is based on operator regression, and it is designed to boost convergence without necessarily increasing the asymptotical error. Operator regression is formulated as a convex quadratically-constrained quadratic programs (QCQPs).

2. We present efficient ways to solve the operator regression problems in dimension $n$ with $\ell$ observations via a pertinent reformulation of the Peaceman-Rachford splitting (PRS) method, see *e.g.* Bauschke and Combettes (2017). Our PRS method is trivially parallel and allows for a reduction of the per-iteration complexity from a convex QCQP in $O(n\ell)$ variables and $O(\ell^2)$ constraints (*i.e.*, a complexity of at least $O((n\ell)^3)$, to $O(\ell^2)$ 1-constraint convex QCQPs in $O(n)$ variables. Importantly, we show that these simpler QCQPs admit a closed form solution, which leads to a per iteration complexity of $O(\ell^2 n)$.

3. We test the performance of the proposed method for two optimization problems: i) a linear regression problem with an ill-conditioned cost [c.f. Mai and Johansson (2020a)]; and, ii) an online phase retrieval problem, which requires the minimization of a weakly convex function [c.f. Duchi and Ruan (2018)]. The proposed operator regression method shows promising performance in both scenarios as compared to forward-backward (with and without backtracking line search) and its accelerated variants FISTA Beck and Teboulle (2009) and an online version of the Anderson acceleration in Mai and Johansson (2020a).

The extended version of this manuscript with appendix and proofs can be found in Bastianello et al. (2021).

### 1.1. Related work

Learning to optimize and regularize is a growing research topic; see Meinhardt et al. (2017); Nghiem et al. (2018); Ongie et al. (2020); Banert et al. (2020); Cohen et al. (2020); Pesquet et al. (2020); Simonetto et al. (2019); Chen et al. (2021) as representative works, even though they focus on slightly different problems. Additional works in the context of learning include the design of convex loss functions in, *e.g.*, Ramaswamy and Agarwal (2016); Finocchiaro et al. (2021). Interpreting algorithms as mappings and operators (averaged, monotone, *etc.*) has been extremely fruitful for characterizing their convergence properties Rockafellar (1976); Eckstein (1989); Bauschke and Combettes (2017); Ryu and Boyd (2016); Sherson et al. (2018).

Convex regression is treated extensively in Seijo and Sen (2011); Lim and Glynn (2012); Mazumder et al. (2019); Blanchet et al. (2019), while recently being generalized to smooth strongly convex functions Simonetto (2021) based on A. Taylor's works Taylor et al. (2017); Taylor (2017); an interesting approach using similar techniques for optimal transport is offered in Paty et al. (2020). Operator regression is a recent and at the same time old topic. We are going to build on the recent work Ryu et al. (2020) and the F.A. Valentine's 1945 paper Valentine (1945).

The Anderson acceleration scheme that we compare with is covered in Mai and Johansson (2020a) (see also Scieur (2018); Zhang et al. (2020)).

And finally, the class of weakly convex functions is broad and important in optimization Rockafellar (1982); Vial (1983); Duchi and Ruan (2018); Davis and Drusvyatskiy (2019); Mai and Johansson (2020b). Applications featuring this class include robust phase retrieval and many others. In control theory, this functions extend, e.g., online identification and control to a class on non-linear dynamical systems, and potential games to hypo-monotone settings.

## 2. Learning to accelerate with operator regression

Consider the time-varying problem (1) and an associated online algorithm $\mathcal{A}_k$, designed to track the optimizers of the problem. The mapping $\mathcal{A}_k$ can be written as sum and/or composition of maps. To fix the ideas and notation, we provide the following example, which will be used throughout the paper to concretely convey ideas (although we note that the proposed methodology is more widely applicable).

**Example.** Consider an online forward-backward type algorithm with updates $\boldsymbol{x}_k = \mathcal{A}_k(\boldsymbol{x}_{k-1})$, where

$$\mathcal{A}_k = \mathrm{prox}_{\alpha g_k} \circ \mathcal{T}_k, \qquad \mathcal{T}_k := \mathcal{I} - \alpha \nabla_{\boldsymbol{x}} f_k \tag{2}$$

where $\mathrm{prox}_{\alpha g_k}$ is the proximal operator ($\mathrm{prox}_{\alpha g_k}(\boldsymbol{y}) = \arg\min_{\boldsymbol{x}}\{g_k(\boldsymbol{x}) + \|\boldsymbol{x}-\boldsymbol{y}\|^2/(2\alpha)\}$) and $\mathcal{I}$ the identity map. The properties of this algorithm depend on the map $\mathcal{T}_k$. In case of a generic smooth non-convex $f_k$ or for convex functions, one can show convergence of the regret to a bounded error Simonetto (2017); Hallak et al. (2020); on the other hand, if $f_k \in \mathcal{S}_{\mu,L}(\mathbb{R}^n)$ uniformly in $k$, $\mu > 0$, then (2) can obtain a linear convergence for the sequence $\{\boldsymbol{x}_k\}$ to the unique optimizer's trajectory of $F_k$ up to a bounded error Dall'Anese et al. (2020). ♦

Our goal can be formulated as follows: if the algorithmic map $\mathcal{A}_k$ is *not* contractive or is only locally contractive, is it possible to find an approximate mapping $\hat{\mathcal{A}}_k$ that is globally contractive to boost the convergence to the optimal solutions (within an error)?

Consider again the proximal-gradient method in the Example, where we recall that $\mathcal{A}_k = \mathrm{prox}_{\alpha g_k} \circ \mathcal{T}_k$, with $\mathcal{T}_k := \mathcal{I} - \alpha \nabla_{\boldsymbol{x}} f_k$. When $f_k$ is $\mu$-strongly convex and $L$-smooth uniformly in time, and $\alpha < 2/L$, the mapping $\mathcal{T}_k$ is contractive; *i.e.*, $\|\mathcal{T}_k(\boldsymbol{x}) - \mathcal{T}_k(\boldsymbol{y})\| \leq \zeta\|\boldsymbol{x} - \boldsymbol{y}\|$ for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and $\zeta \in (0,1)$. Thus, the recursion $\boldsymbol{x}_k = \mathcal{A}_k(\boldsymbol{x}_{k-1})$ achieves linear convergence. However, the question we pose here is the following: when $f_k$ is not $\mu$-strongly convex, can we still learn map $\hat{\mathcal{T}}_k$, and use the surrogate algorithm $\hat{\mathcal{A}}_k = \operatorname{prox}_{\alpha g_k} \circ \hat{\mathcal{T}}_k$ to achieve linear convergence?[2]

To this end, using Fact 2.2 in Ryu et al. (2020), it follows that a mapping $\mathcal{T}_k$ is $\zeta$-contractive interpolable (and therefore extensible to the whole space) if and only if it satisfies:

$$\|\mathcal{T}_k(\boldsymbol{x}_i) - \mathcal{T}_k(\boldsymbol{x}_j)\|^2 \leq \zeta^2 \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2, \forall i, j \in I_\ell,\ i \neq j \tag{3}$$

where $I_\ell := \{1, \dots, \ell\}$ is a finite set of indexes for the points $\{\boldsymbol{x}_i \in \mathbb{R}^n, i \in I_\ell\}$. Therefore, using a number of evaluations $\{\mathcal{T}_k(\boldsymbol{x}_i)\}$ of the mapping $\mathcal{T}_k$ at the points $\{\boldsymbol{x}_i\}$, we pose the following convex QCQP as our operator regression problem:

$$\begin{aligned} \hat{\boldsymbol{t}} = \underset{\mathbb{R}^{n\ell} \ni \boldsymbol{t} = [\boldsymbol{t}_i]_{i \in I_\ell}}{\arg\min} &\ \frac{1}{2} \sum_{i \in I_\ell} \|\boldsymbol{t}_i - \boldsymbol{y}_i\|^2 \\ \text{s.t. } &\ \|\boldsymbol{t}_i - \boldsymbol{t}_j\|^2 \leq \zeta^2 \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2 \ \forall i, j \in I_\ell, i \neq j, \end{aligned} \tag{4}$$

where the cost function represents a least-square criterion on the "observations" (*i.e.*, the evaluations of the mapping) $\boldsymbol{y}_i := \mathcal{T}_k(\boldsymbol{x}_i)$, $i \in I_\ell$, and the constraints enforce contractivity. In particular, the optimal values $\hat{\boldsymbol{t}}$ on the data points represent the evaluations of a $\zeta$-contracting operator when applied to those points, *i.e.*, $\hat{\boldsymbol{t}}_i = \hat{\mathcal{T}}_k(\boldsymbol{x}_i)$.

### 2.1. PRS-based solver

The convex problem (4) can be solved using off-the-shelf solvers for convex programs; however, the computational complexity may be a limiting factor, since the problem has a number of constraints that scales quadratically with the number of data points $\ell$. In particular, the computational complexity of interior-point methods would scale at least as $O((n\ell)^3)$. This is generally the case in non-parametric regression Mazumder et al. (2019); Aybat and Wang (2014). To resolve this issue, we propose a parallel algorithm that solves (4) more efficiently based on the so-called Peaceman-Rachford splitting (PRS), see *e.g.* Bauschke and Combettes (2017), and that leverages the closed form solution of particular 1-constraint QCQPs.

To this end, define the following set of pairs $\mathcal{V} = \{e = (i,j) \mid i, j \in I_\ell,\ i < j\}$ which are ordered (that is, for example we take $(1,2)$ and not $(2,1)$, to avoid counting the pair twice). We associate with each pair $e = (i,j)$ the constraint $\|\boldsymbol{t}_i - \boldsymbol{t}_j\|^2 \leq \zeta^2 \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2$, for a total of $\ell(\ell-1)/2$ constraints.

Let $\boldsymbol{t}_{i,e}$ and $\boldsymbol{t}_{j,e}$ be copies of $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$ associated to the $e$-th pair; then we can equivalently reformulate problem (4) as

$$\min_{\boldsymbol{t}_{i,e}, \boldsymbol{t}_{j,e}} \frac{1}{2(\ell-1)} \sum_{e \in \mathcal{V}} \left\| \begin{bmatrix} \boldsymbol{t}_{i,e} \\ \boldsymbol{t}_{j,e} \end{bmatrix} - \begin{bmatrix} \boldsymbol{y}_i \\ \boldsymbol{y}_j \end{bmatrix} \right\|^2 \quad \text{s.t.} \ \begin{array}{l} \|\boldsymbol{t}_{i,e} - \boldsymbol{t}_{j,e}\|^2 \leq \zeta^2 \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2 \\ \boldsymbol{t}_{i,e} = \boldsymbol{t}_{i,e'} \ \forall e, e' | i \sim e, e' \end{array}, \tag{5}$$

where we write that $i \sim e$ if the $e$-th constraint involves $\boldsymbol{t}_i$, and recall that $\boldsymbol{y}_i := \mathcal{T}_k(\boldsymbol{x}_i)$, $i \in I_\ell$, $i \in I_\ell$. Problem (5) is a strongly convex problem with convex constraints defined in the variables $\boldsymbol{t}_{i,e}$. Problem (5) is in fact a consensus problem which can be decomposed over the pairs $\mathcal{V}$ by using PRS, as defined in the following lemma.

[2] Notice that the proximal of $g_k$ – which may encode important properties such as sparsity or constraints – is not subjected to the learning procedure and remains unchanged.

**Lemma 1** *Problem (5) can be solved by using Peaceman-Rachford splitting (PRS), yielding the following iterative procedure. Given the penalty $\rho > 0$, apply for $h \in \mathbb{N}$:*

$$\begin{bmatrix} \boldsymbol{t}_{i,e}^h \\ \boldsymbol{t}_{j,e}^h \end{bmatrix} = \operatorname*{arg\,min}_{\boldsymbol{t}_{i,e}, \boldsymbol{t}_{j,e}} \left\{ \frac{1}{2(\ell-1)} \left\| \begin{bmatrix} \boldsymbol{t}_{i,e} \\ \boldsymbol{t}_{j,e} \end{bmatrix} - \begin{bmatrix} \boldsymbol{y}_i \\ \boldsymbol{y}_j \end{bmatrix} \right\|^2 + \frac{1}{2\rho} \left\| \begin{bmatrix} \boldsymbol{t}_{i,e} \\ \boldsymbol{t}_{j,e} \end{bmatrix} - \boldsymbol{z}_e^h \right\|^2 \right\} \tag{6a}$$
$$\text{s.t.} \quad \|\boldsymbol{t}_{i,e} - \boldsymbol{t}_{j,e}\|^2 \leq \zeta^2 \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2$$

$$\boldsymbol{v}_{i,e}^h = \frac{1}{\ell-1} \sum_{e' | i \sim e'} \left( 2\boldsymbol{t}_{i,e'}^h - \boldsymbol{z}_{e',i}^h \right), \qquad \boldsymbol{z}_e^{h+1} = \boldsymbol{z}_e^h + \begin{bmatrix} \boldsymbol{v}_{i,e}^h - \boldsymbol{t}_{i,e}^h \\ \boldsymbol{v}_{j,e}^h - \boldsymbol{t}_{j,e}^h \end{bmatrix}. \tag{6b}$$

*At each iteration, the algorithm solves in parallel $\ell(\ell-1)/2$ convex QCQPs – each in $2n$ variables and $1$ constraint – and then aggregates the results. Importantly, the following lemma shows that 1-constraint QCQPs can be solved in closed form with a complexity of $O(n)$, and hence the total per iteration complexity of (6) is $O(\ell^2 n)$.*

**Lemma 2 (Solving 1-constraint QCQPs)** *Consider the (prototypical) QCQP with one constraint*

$$(\boldsymbol{t}_i^*, \boldsymbol{t}_j^*) = \arg\min \frac{1}{2} \left\| \begin{bmatrix} \boldsymbol{t}_i \\ \boldsymbol{t}_j \end{bmatrix} - \begin{bmatrix} \boldsymbol{w}_i \\ \boldsymbol{w}_j \end{bmatrix} \right\|^2 \qquad \text{s.t.} \quad \frac{1}{2} \|\boldsymbol{t}_i - \boldsymbol{t}_j\|^2 - b \leq 0 \tag{7}$$

*where $b > 0$, which includes as a particular case the update (6a). Problem (7) admits the following closed form solution*

$$\lambda^* = \max\left\{0, \frac{1}{2}\left(\frac{\|\boldsymbol{w}_i - \boldsymbol{w}_j\|}{\sqrt{2b}} - 1\right)\right\}, \qquad \begin{bmatrix} \boldsymbol{t}_i^* \\ \boldsymbol{t}_j^* \end{bmatrix} = \frac{1}{1+2\lambda^*} \left( \begin{bmatrix} 1+\lambda^* & \lambda^* \\ \lambda^* & 1+\lambda^* \end{bmatrix} \otimes \boldsymbol{I}_n \right) \begin{bmatrix} \boldsymbol{w}_i \\ \boldsymbol{w}_j \end{bmatrix}. \tag{8}$$

Leveraging Lemma 2, we see that (6a) has the following closed form solution

$$\begin{bmatrix} \boldsymbol{w}_{i,e}^h \\ \boldsymbol{w}_{j,e}^h \end{bmatrix} = \frac{1}{\ell-1+\rho} \left( \rho \begin{bmatrix} \boldsymbol{y}_i \\ \boldsymbol{y}_j \end{bmatrix} + (\ell-1)\boldsymbol{z}_e^h \right), \quad \begin{bmatrix} \boldsymbol{t}_{i,e}^h \\ \boldsymbol{t}_{j,e}^h \end{bmatrix} = \frac{1}{1+2\lambda_e^h} \left( \begin{bmatrix} 1+\lambda_e^h & \lambda_e^h \\ \lambda_e^h & 1+\lambda_e^h \end{bmatrix} \otimes \boldsymbol{I}_n \right) \begin{bmatrix} \boldsymbol{w}_{i,e}^h \\ \boldsymbol{w}_{j,e}^h \end{bmatrix}, \tag{9a}$$

$$\text{with } \lambda_e^h = \max\left\{0, \frac{1}{2}\left(\frac{\left\|\boldsymbol{w}_{i,e}^h - \boldsymbol{w}_{j,e}^h\right\|}{\zeta \|\boldsymbol{x}_i - \boldsymbol{x}_j\|} - 1\right)\right\}. \tag{9b}$$

Finally, we discuss how the closed form solution of 1-constraint QCQPs leads to a very low per iteration complexity.

**Lemma 3 (Computational complexity)** *Consider the Peaceman-Rachford splitting (6) that solves the operator regression problem (5), and further notice that the 1-constraint QCQPs (6a) have a closed form solution described in Lemma 2.*

*Then, the computational complexity of the PRS solver is $O(\ell^2 n)$ per iteration. In particular, when the budget of operator calls $\ell$ is much smaller than the dimension of the problem ($n \gg \ell$), then the complexity reduces to $O(n)$ per iteration.*

## 3. OpReg-Boost

We are now ready to present our main algorithm. To convey ideas concretely, we focus here on online algorithms of the forward-backward type as in Example 1, *i.e.*,[3]

$$\boldsymbol{x}_k = \mathrm{prox}_{\alpha g_k}\left(\mathcal{T}_k(\boldsymbol{x}_{k-1})\right), \quad k \in \mathbb{N},\ \alpha > 0. \tag{10}$$

where we recall that $\mathcal{T}_k = \mathcal{I} - \alpha \nabla_{\boldsymbol{x}} f_k$. In particular, we will utilize the operator regression method on the mapping $\mathcal{T}_k$. We recall that the prox operator is non-expansive; therefore, the Lipschitz constant of the overall mapping $\mathrm{prox} \circ \mathcal{T}_k$ depends on the mathematical properties of $\mathcal{T}_k$ Bauschke et al. (2012) and, more specifically, of the function $f_k$. In particular, since $f_k$ is not assumed to be strongly convex in general, $\mathcal{T}_k$ may not be contractive and, consequently, $\mathrm{prox} \circ \mathcal{T}_k$ is not contractive either. With this in mind, the goal is to learn a contracting mapping $\hat{\mathcal{T}}_k$ from evaluations of $\mathcal{T}_k$ at some points. The OpReg-Boost algorithm can thus be described as follows.

**OpReg-Boost algorithm**

```
Required: number of points $\ell$, stepsize $\alpha$, contraction factor $\zeta$, initial condition $\boldsymbol{x}_0$.
At each time $k$ do:
[S1] Learn the closest contracting operator to $\mathcal{T}_k$, say $\hat{\mathcal{T}}_k$ by:
    [S1.1] Choose $\ell - 1$ points $\{\boldsymbol{x}_p\}$ around $\boldsymbol{x}_{k-1}$ to create the set of points $\{\boldsymbol{x}_i\} := \{\boldsymbol{x}_{k-1} \cup \{\boldsymbol{x}_p\}\}$, $i \in I_\ell$, where the map $\mathcal{T}_k$ is to be evaluated.
    [S1.2] Evaluate the mapping at the data points: $\boldsymbol{y}_i = \mathcal{T}_k(\boldsymbol{x}_i)$, $i \in I_\ell$, i.e., $\boldsymbol{y}_i = \boldsymbol{x}_i - \alpha \nabla_{\boldsymbol{x}} f_k(\boldsymbol{x}_i)$.
    [S1.3] Solve (4) on $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}$, $i \in I_\ell$ with the PRS-based algorithm.
    [S1.4] Output $\hat{\boldsymbol{t}}_k (= \hat{\mathcal{T}}_k(\boldsymbol{x}_{k-1}))$ from the solution of [S1.3].
[S2] Compute $\boldsymbol{x}_k = \mathrm{prox}_{\alpha g_k}(\hat{\boldsymbol{t}}_k)$.
```

A couple of remarks are in order. First, the computational complexity of the overall algorithm is dominated by the operation regression problem (4) in step [S1.3]; on the other hand, the number of gradient calls (used to evaluate $\mathcal{T}_k$) is $\ell$-times the one of a standard forward-backward algorithm. At each time $k$, we perform $\ell$ gradient evaluations at the points $\boldsymbol{x}_{k-1} \cup \{\boldsymbol{x}_p\}$ (the points $\{\boldsymbol{x}_p\}$ could be obtained, *e.g.*, by adding a zero-mean Gaussian noise term to $\boldsymbol{x}_{k-1}$). $\ell$ can be as small as 3 in practice.

Second, as one can see from steps [S1.2]–[S1.4], the operation regression problem (4) directly provides the evaluation of the regularized operator at the data point $\boldsymbol{x}_{k-1}$, since we choose $\boldsymbol{x}_{k-1}$ to define one of the training points.

## 4. Numerical results

We present a number of experiments to evaluate the performance of the proposed method[4]. We consider: *(i)* an ill-conditioned online linear regression with a convex cost (but not strongly convex); *(ii)* an online phase retrieval problem that is weakly convex, and which is characterized by a high computational cost per operator evaluation. The first example is rather well-studied, at least

[3] Access to an operator is the only requirement for the application of OpReg-Boost. However, it is instructive to fix the ideas on a concrete mapping by focusing on the forward-backward algorithm.

[4] The experiments were implemented in Python and performed on a computer with Intel i7-4790 CPU, 3.60GHz, and 8GB of RAM, running Linux. Code and data are available. The implementation is serial (possible due to the manageable size of the regression problems); future work will look at parallel implementations.

in the well-conditioned region, and it can be used for example to derive control laws under sparsity requirements Ohlsson et al. (2010); Lin et al. (2013). The second example has important repercussions in adaptive optics, where phase retrieval techniques are used as building blocks to generate control signals Massioni et al. (2011); Antonello and Verhaegen (2015)

The metric used in the experiments is the tracking error, characterized as the distance from the ground truth signal $\boldsymbol{y}_k$ of the solution output by the solvers. By $\boldsymbol{y}_k$ we denote the signal being tracked via linear regression in section 4.1 or the phase being retrieved in section 4.2. We choose the tracking error as a proxy for the distance to the optimizer $\boldsymbol{x}_k^*$ in line with the work of Duchi and Ruan (2018), since *(i)* determining $\boldsymbol{x}_k^*$ is in general hard to do computationally in the problems we are considering and it may not be unique, and *(ii)* the tracking error very naturally provides insights on how the methods perform in estimating the real signals.

### 4.1. Online linear regression

We consider the following time-varying problem:

$$\boldsymbol{x}_k^* \in \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathbb{R}^n} \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}_k\|^2 + w \|\boldsymbol{x}\|_1 , \tag{11}$$

with $n = 1000$, $w = 1000$, $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ such that $\operatorname{rank}(\boldsymbol{A}) = n/2$ and having maximum and minimum (non-zero) eigenvalues $\sqrt{L}$, $\sqrt{\mu}$. The goal is to reconstruct a signal $\boldsymbol{y}_k$ with sinusoidal components, $1/3$ of them being zero, from the noisy observations $\boldsymbol{b}_k = \boldsymbol{A}\boldsymbol{y}_k + \boldsymbol{e}_k$, and $\boldsymbol{e}_k \sim \mathcal{N}(\boldsymbol{0}, 10^{-2}\boldsymbol{I})$. Due to $\boldsymbol{A}$ being rank deficient, the cost $f_k$ is convex but not strongly so, and we have $\lambda_{\max}(\nabla_{\boldsymbol{x}\boldsymbol{x}} f_k)/\tilde{\lambda}_{\min}(\nabla_{\boldsymbol{x}\boldsymbol{x}} f_k) = L/\mu$, where $\lambda_{\max}$ and $\tilde{\lambda}_{\min}$ are the maximum and minimum non-zero eigenvalues of a matrix. The function $f_k$ changes every $\delta = 0.1s$.

In Figure 2, we show a comparison of the tracking error attained by the proposed OpReg-Boost against the forward-backward method, and its accelerated versions FISTA (with and without backtracking line search) Beck and Teboulle (2009), and (guarded) Anderson Mai and Johansson (2020a). The methods are given the same computational time budget[5], the step-size of forward-backward is $\alpha = 2/(L + \mu)$, and the parameters of OpReg-Boost are $\ell = 3$ and $\rho = 10^{-6}$. For large values of $L$ OpReg-Boost outperforms all other methods; in the case $L = 10^4$ it performs slightly worse in terms of asymptotic error, but successfully improves the convergence rate. The reason behind the performance we observe is that as $L$ grows larger, the allowed step-size for forward-backward becomes smaller – indeed, we have the bound $\alpha < 2/L$. We further remark that the performance of OpReg-Boost can be improved in the case $L = 10^4$ by choosing a different value of $\rho$, see Bastianello et al. (2021).

Finally, in Table 1 we report the asymptotic error and computational time of OpReg-Boost as compared to the forward-backward based solvers for three different sizes of the problem with $L = 10^8$ and $\mu = 1$. In terms of asymptotic error – evaluated when all methods are given the same total computational time – the performance of OpReg-Boost is consistently better than the other methods. Regarding the computational time per step of the algorithm we see that OpReg-Boost is comparable with the accelerated methods FISTA with backtracking and Anderson. On the

[5] Specifically, we evaluate the computational time required by one iteration of OpReg-Boost, and run the other methods for the same time. We remark that OpReg-Boost requires at least the time needed by $\ell$ iterations of forward-backward to generate the operator regression data. For example, our experiments show that with the choice $\rho = 10^{-6}$ during one iteration of OpReg-Boost we can apply $\ell + 1$ of forward-backward or FISTA, and one or two of Anderson and FISTA with backtracking.

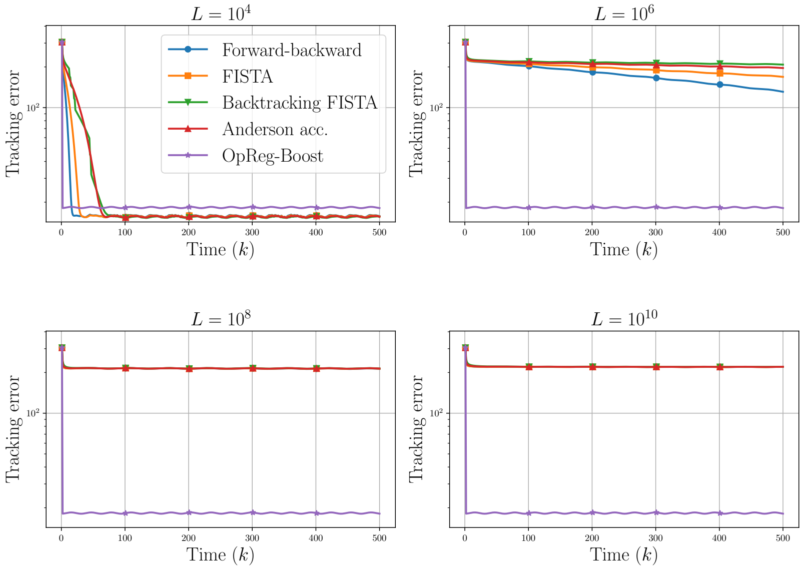

Figure 2: Comparison with a fixed computational time budget per time $k \in \mathbb{N}$, and for different values of $L$, with fixed $\mu = 1$.

Table 1: Comparison for different values of $n$; for each algorithm we report the asymptotic error (as. err.) and the average computational time **per step of the algorithm** (t. / s.). We remark that in the simulations **all methods are given the same computational time budget**, so we apply one or more steps of the algorithm. The simulations are for $L = 10^8$ and $\mu = 1$.

| | $n = 10$ | | $n = 100$ | | $n = 1000$ | |
|---|---|---|---|---|---|---|
| Algorithm | as. err. | t. / s. [s] | as. err. | t. / s. [s] | as. err. | t. / s. [s] |
| Forward-backward | 30.00 | $3.76 \times 10^{-5}$ | 64.88 | $3.91 \times 10^{-5}$ | 221.36 | $8.55 \times 10^{-4}$ |
| FISTA | 29.69 | $3.44 \times 10^{-5}$ | 62.15 | $4.20 \times 10^{-5}$ | 221.00 | $8.41 \times 10^{-4}$ |
| FISTA (backtr.) | 29.69 | $6.33 \times 10^{-4}$ | 62.15 | $8.27 \times 10^{-4}$ | 220.98 | $1.77 \times 10^{-2}$ |
| Anderson | 29.69 | $1.07 \times 10^{-4}$ | 62.16 | $1.27 \times 10^{-4}$ | 221.01 | $1.71 \times 10^{-3}$ |
| **OpReg-Boost** | **2.11** | $\mathbf{2.48 \times 10^{-4}}$ | **6.14** | $\mathbf{2.98 \times 10^{-4}}$ | **18.72** | $\mathbf{2.88 \times 10^{-3}}$ |

other hand, the computationally lighter forward-backward and FISTA require less time per step, but, again, when given the same computational time the performance of OpReg-Boost is still better.

## 4.2. Online phase retrieval

We consider now the following phase retrieval problem presented in Duchi and Ruan (2018):

$$\boldsymbol{x}_k^* \in \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} \left| \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle^2 - b_{i,k} \right|, \tag{12}$$

where the goal is to reconstruct the time-varying signal $\boldsymbol{y}_k \in \mathbb{S}^{n-1}$, $n = 50$, from the noisy measurements $b_{i,k} = \langle \boldsymbol{a}_i, \boldsymbol{y}_k \rangle + \xi_{i,k}$, $i = 1, \ldots, m$ and $m = 100$. The signal $\boldsymbol{y}_k$ is piece-wise constant, with the value of each constant piece being independently drawn. The additive noises are i.i.d. Laplace with zero mean and scale parameter 1. The $\boldsymbol{a}_i$ are the rows of $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, constructed as $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{D}$, with $\boldsymbol{U} \in \mathbb{R}^{m \times n}$ an orthogonal matrix, and $\boldsymbol{D} \in \mathbb{R}^{n \times n}$ a diagonal one with elements

$L = 10^2$, $\mu = 1$ (hence the condition number of $\boldsymbol{A}$ is $10^2$), and the remaining $n-2$ drawn from $\mathcal{U}[\mu, L]$. The problem changes every $\delta = 1s$.

We consider the *prox-linear* solver proposed in the work of Drusvyatskiy and Lewis (2018) (see also Duchi and Ruan (2018)), characterized by $\mathcal{T}_k(\boldsymbol{y}) = \mathrm{prox}_{\alpha f_{k,\boldsymbol{y}}}(\boldsymbol{y})$, where $f_{k,\boldsymbol{y}}$ denotes the following linearized version of the cost in (12): $f_{k,\boldsymbol{y}}(\boldsymbol{x}) = \frac{1}{m}\sum_{i=1}^{m}\left|\langle \boldsymbol{a}_i, \boldsymbol{y}\rangle^2 + 2\langle \boldsymbol{a}_i, \boldsymbol{y}\rangle\langle \boldsymbol{a}_i, \boldsymbol{x}-\boldsymbol{y}\rangle - b_{i,k}\right|$. We choose the step-size of the prox-linear solver as $\alpha = 10^{-3}$, which empirically led to convergence (at least in the initial transient) without the need for line search. Notice that the proximal operator $\mathcal{T}_k$ does not have a closed form, and each operator call requires the solution of a quadratic program, which takes $0.177 \pm 0.0052s$.

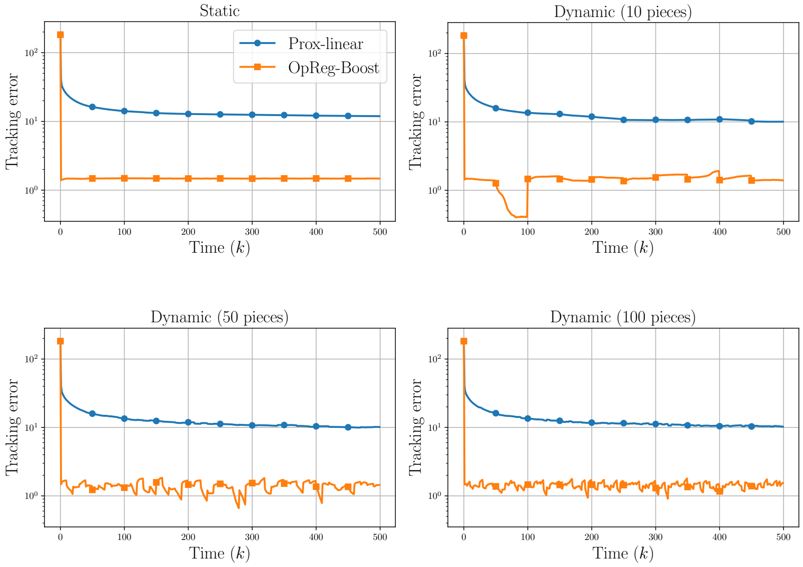

Figure 3: Comparison of the tracking error evolution for prox-linear Drusvyatskiy and Lewis (2018) and OpReg-Boost. The methods are tasked with retrieving phase signals that are piecewise continuous, with different number of pieces in each.

We also consider our OpReg-Boost algorithm applied to the operator[6] $\mathrm{proj}_{\mathbb{S}^{n-1}} \circ \mathcal{T}_k$, and which regularizes the operator $\mathcal{T}_k$ to a $\zeta$-contractive operator, yielding then $\mathrm{proj}_{\mathbb{S}^{n-1}} \circ \hat{\mathcal{T}}_k$. The solution of an operator regression problem requires $1.35 \times 10^{-3} \pm 0.011s$. In the results, then, during the time before the arrival of a new problem ($\delta = 1s$), we perform either 4 steps of the prox-linear solver, or one step of OpReg-Boost with 3 training points (and choosing the PRS parameter $\rho = 10^{-4}$).

In Figure 3, we show the tracking error of prox-linear compared with OpReg-Boost when the signal has different numbers of constant pieces (from being static – one constant value – to being highly dynamic – changing every $5s$). As we can see, OpReg-Boost consistently outperforms prox-linear, including in the static case, in which OpReg-Boost quickly converges to the (approximate) fixed point, while prox-linear converges more slowly.

[6] This shows better performance in practice rather than regularizing $\mathcal{T}_k$ alone. Strictly speaking, with this choice, function $g_k$ in (1) would be the indicator function of a non-convex set. The good performance of the proposed approach however suggests that it can be applied to more general problems than (1).

## Acknowledgments

The work of Nicola Bastianello was supported in part by the Research Project PRIN 2017 Advanced Network Control of Future Smart Grids funded by the Italian Ministry of University and Research (2020-2023) http://vectors.dieti.unina.it and partially by MIUR (Italian Minister for Education) under the initiative "Departments of Excellence" (Law 232/2016).

## References

Jacopo Antonello and Michel Verhaegen. Modal-based phase retrieval for adaptive optics. *J. Opt. Soc. Am. A*, 32(6):1160–1170, Jun 2015.

M. S. Asif and J. Romberg. Sparse recovery of streaming signals using $\ell_1$-homotopy . *IEEE Transactions on Signal Processing*, 62(16):4209 – 4223, 2014.

N. S. Aybat and Z. Wang. A Parallel Method for Large Scale Convex Regression Problems. In *Proceedings of the IEEE Conference in Decision and Control*, 2014.

Sebastian Banert, Axel Ringh, Jonas Adler, Johan Karlsson, and Ozan Öktem. Data-Driven Nonsmooth Optimization. *SIAM Journal on Optimization*, 30(1):102–131, 2020.

N. Bastianello, A. Simonetto, and R. Carli. Distributed Prediction-Correction ADMM for Time-Varying Convex Optimization. In *Proceedings of the Asilomar Conference on Signals, Systems, and Computers*, 2020.

Nicola Bastianello, Andrea Simonetto, and Emiliano Dall'Anese. OpReg-Boost: Learning to Accelerate Online Algorithms with Operator Regression. *arXiv:2105.13271 [cs, math]*, July 2021. URL http://arxiv.org/abs/2105.13271.

Heinz H. Bauschke and Patrick L. Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. CMS books in mathematics. Springer, Cham, 2 edition, 2017.

Heinz H. Bauschke, Sarah M. Moffat, and Xianfu Wang. Firmly Nonexpansive Mappings and Maximally Monotone Operators: Correspondence and Duality. *Set-Valued and Variational Analysis*, 20(1):131–153, March 2012.

Amir Beck and Marc Teboulle. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, January 2009.

Giuseppe Belgioioso, Angelia Nedić, and Sergio Grammatico. Distributed generalized nash equilibrium seeking in aggregative games on time-varying networks. *IEEE Transactions on Automatic Control*, 66(5):2061–2075, 2021.

F. Berkenkamp, R. Moriconi, A. P. Schoellig, and A. Krause. Safe learning of regions of attraction for uncertain, nonlinear systems with Gaussian processes. In *Proceedings of the 55th Conference on Decision and Control*, pages 4661 – 4666, December 2016.

O. Besbes, Y. Gur, and A. Zeevi. Non-stationary Stochastic Optimization. *Operations research*, 63 (5):1227 – 1244, 2015.

J. Blanchet, P. W. Glynn, J. Yan, and Z. Zhou. Multivariate Distributionally Robust Convex Regression under Absolute Error Loss. In *Proceedings of NeurIPS*, 2019.

T. Chen, X. Chen, W. Chen, H. Heaton, J. Liu, Z. Wang, and W. Yin. Learning to optimize: A primer and a benchmark. *arXiv preprint arXiv:2103.12828*, 2021.

R. Cohen, M. Elad, and P. Milanfar. Regularization by Denoising via Fixed-Point Projection (RED-PRO). *arXiv:2008.00226*, 2020.

E. Dall'Anese, A. Simonetto, S. Becker, and L. Madden. Optimization and Learning with Information Streams: Time-varying Algorithms and Applications. *IEEE Signal Processing Magazine*, May 2020.

D. Davis and D. Drusvyatskiy. Stochastic Model-Based Minimization of Weakly Convex Functions. *SIAM Journal on Optimization*, 29(1):207–239, 2019.

O. Devolder, F. Glineur, and Yu. Nesterov. Double Smoothing Technique for Large-Scale Linearly Constrained Convex Optimization. *SIAM Journal on Optimization*, 22(2):702 – 727, 2012.

Dmitriy Drusvyatskiy and Adrian S. Lewis. Error Bounds, Quadratic Growth, and Linear Convergence of Proximal Methods. *Mathematics of Operations Research*, 43(3):919–948, August 2018.

John C. Duchi and Feng Ruan. Stochastic Methods for Composite and Weakly Convex Optimization Problems. *SIAM Journal on Optimization*, 28(4):3229–3259, January 2018.

J. Eckstein. *Splitting Methods for Monotone Operators with Applications to Parallel Optimization*. PhD thesis, MIT, June 1989.

Filippo Fabiani, Andrea Simonetto, and Paul J. Goulart. Learning equilibria with personalized incentives in a class of nonmonotone games. *arXiv:2111.03854 [cs, eess, math]*, November 2021.

Jessie Finocchiaro, Rafael Frongillo, and Bo Waggoner. Unifying lower bounds on prediction dimension of consistent convex surrogates. *arXiv preprint arXiv:2102.08218*, 2021.

Eric C Hall and Rebecca M Willett. Online convex optimization in dynamic environments. *IEEE Journal of Selected Topics in Signal Processing*, 9(4):647–662, 2015.

Nadav Hallak, Panayotis Mertikopoulos, and Volkan Cevher. Regret minimization in stochastic non-convex learning via a proximal-gradient approach. *arXiv preprint arXiv:2010.06250*, 2020.

A. Jadbabaie, A. Rakhlin, S. Shahrampour, and K. Sridharan. Online Optimization: Competing with Dynamic Comparators. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, PMLR*, number 38, pages 398 – 406, 2015.

J. Koshal, A. Nedić, and U. Y. Shanbhag. Multiuser Optimization: Distributed Algorithms and Error Analysis. *SIAM Journal on Optimization*, 21(3):1046 – 1081, 2011.

Y. Li, G. Qu, and N. Li. Online Optimization with Predictions and Switching Costs: Fast Algorithms and the Fundamental Limit. *arXiv: 1801.07780*, 2020.

Dominic Liao-McPherson, Marco Nicotra, and Ilya Kolmanovsky. A Semismooth Predictor Corrector Method for Real-Time Constrained Parametric Optimization with Applications in Model Predictive Control. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 3600–3607, December 2018.

E. Lim and P. W. Glynn. Consistency of Multidimensional Convex Regression. *Operation Research*, 60(1):196 – 208, 2012.

Fu Lin, Makan Fardad, and Mihailo R. Jovanovic. Design of optimal sparse feedback gains via the alternating direction method of multipliers. *IEEE Transactions on Automatic Control*, 58(9): 2426–2431, 2013.

X. Luo, Y. Zhang, and M. M. Zavlanos. Socially-Aware Robot Planning via Bandit Human Feedback. In *2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*, pages 216–225, 2020.

V. Mai and M. Johansson. Anderson Acceleration of Proximal Gradient Methods. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6620–6629, 2020a.

V. Mai and M. Johansson. Convergence of a Stochastic Gradient Method with Momentum for Non-Smooth Non-Convex Optimization. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6630–6639, 2020b.

Paolo Massioni, Caroline Kulcsár, Henri-François Raynaud, and Jean-Marc Conan. Fast computation of an optimal controller for large-scale adaptive optics. *Journal of the Optical Society of America. A Optics, Image Science, and Vision*, 28(11):2298–2309, 2011.

R. Mazumder, A. Choudhury, G. Iyengar, and B. Sen. A Computational Framework for Multivariate Convex Regression and Its Variants. *Journal of the American Statistical Association*, 114(525): 318–331, 2019.

T. Meinhardt, M. Moller, C. Hazirbas, and D. Cremers. Learning proximal operators: Using denoising networks for regularizing inverse imaging problems. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1781–1790, 2017.

Aryan Mokhtari, Shahin Shahrampour, Ali Jadbabaie, and Alejandro Ribeiro. Online optimization in dynamic environments: Improved regret rates for strongly convex problems. In *IEEE Conference on Decision and Control*, pages 7195–7201, 2016.

Yu. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1): 127–152, 2005.

T.X. Nghiem, G. Stathopoulos, and C.N. Jones. Learning Proximal Operators with Gaussian Processes. In *Proceedings of the Allerton Conference on Communication, Control, and Computing*, 2018.

Henrik Ohlsson, Fredrik Gustafsson, Lennart Ljung, and Stephen Boyd. Trajectory generation using sum-of-norms regularization. In *49th IEEE Conference on Decision and Control (CDC)*, pages 540–545, 2010.

G. Ongie, A. Jalal, R.G. Baraniuk C.A. Metzler, A.G. Dimakis, and R. Willett. Deep Learning Techniques for Inverse Problems in Imaging. *IEEE Journal on Selected Areas in Information Theory*, 5, 2020.

Santiago Paternain, Manfred Morari, and Alejandro Ribeiro. A Prediction-Correction Method for Model Predictive Control. In *2018 Annual American Control Conference (ACC)*, pages 4189–4194, June 2018.

François-Pierre Paty, Alexandre d'Aspremont, and Marco Cuturi. Regularity as Regularization: Smooth and Strongly Convex Brenier Potentials in Optimal Transport. In *Proceedings of AISTATS*, 2020.

J.-C. Pesquet, A. Repetti, M. Terris, and Y. Wiaux. Learning Maximally Monotone Operators for Image Recovery. *arXiv:2012.13247*, 2020.

A Yu Popkov. Gradient methods for nonstationary unconstrained optimization problems. *Automation and Remote Control*, 66(6):883–891, 2005.

Harish G Ramaswamy and Shivani Agarwal. Convex calibration dimension for multiclass loss matrices. *The Journal of Machine Learning Research*, 17(1):397–441, 2016.

R. T. Rockafellar. Monotone Operators and the Proximal Point Algorithm. *SIAM Journal of Control and Optimization*, 14(5):877 – 898, 1976.

T. R. Rockafellar. Favorable classes of Lipschitz continuous functions in subgradient optimization. *In Nurminski, E. A. (ed.), Progress in Nondifferentiable Optimization*, pages 125 – 143, 1982.

E. K. Ryu and S. Boyd. Primer on Monotone Operator Methods. *Applied Computational Mathematics*, 15(1):3 – 43, 2016.

Ernest K. Ryu, Adrien B. Taylor, Carolina Bergeling, and Pontus Giselsson. Operator Splitting Performance Estimation: Tight Contraction Factors and Optimal Parameter Selection. *SIAM Journal on Optimization*, 30(3):2251–2271, 2020.

D. Scieur. *Acceleration in Optimization*. PhD thesis, PSL Research University, France, September 2018.

E. Seijo and B. Sen. Nonparametric Least Squares Estimation of a Multivariate Convex Regression Function. *The Annals of Statistics*, 39(3):1633 – 1657, 2011.

T. Sherson, R. Heusdens, and W.B. Kleijn. Derivation and Analysis of the Primal-Dual Method of Multipliers Based on Monotone Operator Theory. *IEEE Transactions on Signal and Information Processing over Networks*, 5(2):334–347, 2018.

A. Simonetto. Time-Varying Convex Optimization via Time-Varying Averaged Operators . *arXiv: 1704.07338v1*, 2017.

A. Simonetto and G. Leus. Double Smoothing for Time-Varying Distributed Multi-user Optimization. In *Proceedings of the IEEE Global Conference on Signal and Information Processing*, Atlanta, US, December 2014.

Andrea Simonetto. Smooth Strongly Convex Regression. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 2130–2134, Amsterdam, January 2021. IEEE.

Andrea Simonetto, Emiliano Dall'Anese, Julien Monteil, and Andrey Bernstein. Personalized optimization with user's feedback. *arXiv preprint arXiv:1905.00775*, 2019.

A. Taylor. *Convex Interpolation and Performance Estimation of First-order Methods for Convex Optimization*. PhD thesis, Université catholique Louvain, Belgium, January 2017.

A.B. Taylor, J.M. Hendrickx, and F. Glineur. Smooth Strongly Convex Interpolation and Exact Worst-case Performance of First-order Methods. *Mathematical Programming*, 161(1):307 – 345, 2017.

Frederick Albert Valentine. A Lipschitz condition preserving extension for a vector function. *American Journal of Mathematics*, 67(1):83–93, 1945.

J.-P. Vial. Strong and weak convexity of sets and functions. *Mathematics of Operations Research*, 8(2):231–259, 1983.

Junzi Zhang, Brendan O'Donoghue, and Stephen Boyd. Globally Convergent Type-I Anderson Acceleration for Nonsmooth Fixed-Point Iterations. *SIAM Journal on Optimization*, 30(4):3170–3197, 2020.

Runyu Zhang, Yingying Li, and Na Li. On the Regret Analysis of Online LQR Control with Predictions. In *2021 American Control Conference (ACC)*, pages 697–703, 2021.

Yang Zheng and Na Li. Non-Asymptotic Identification of Linear Dynamical Systems Using Multiple Trajectories. *IEEE Control Systems Letters*, 5(5):1693–1698, 2021.