Prediction-Correction Algorithms for Time-Varying Constrained Optimization

Andrea Simonetto and Emiliano Dall'Anese

Abstract—This paper develops online algorithms to track solutions of time-varying constrained optimization problems. Particularly, resembling workhorse Kalman filtering-based approaches for dynamical systems, the proposed methods involve prediction-correction steps to provably track the trajectory of the optimal solutions of time-varying convex problems. The merits of existing prediction-correction methods have been shown for unconstrained problems and for setups where computing the inverse of the Hessian of the cost function is computationally affordable. This paper addresses the limitations of existing methods by tackling constrained problems and by designing first-order prediction steps that rely on the Hessian of the cost function (and do not require the computation of its inverse). In addition, the proposed methods are shown to improve the convergence speed of existing prediction-correction methods when applied to unconstrained problems. Numerical simulations corroborate the analytical results and showcase performance and benefits of the proposed algorithms. A realistic application of the proposed method to real-time control of energy resources is presented.

Index Terms—Time-varying optimization, non-stationary optimization, parametric programming, prediction-correction methods, real-time control of energy resources.

I. Introduction

We consider constrained optimization problems that vary continuously in time. We outline the main ideas by first focusing on problems with a time-varying objective function. Consider then the following optimization problem:

$$\boldsymbol{x}^*(t) := \operatorname*{argmin}_{\boldsymbol{x} \in X} f(\boldsymbol{x}; t), \quad \text{for } t \geqslant 0, \tag{1}$$

where $X \subseteq \mathbb{R}^n$ is a convex set; $t \in \mathbb{R}_+$ is non-negative, continuous, and it is used to index time; and, $f : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}$ is a *smooth strongly convex function*. The goal is to find (and track) the solution $\boldsymbol{x}^*(t)$ of (1) for each time t – hereafter referred to as the optimal solution *trajectory*.

Problem (1) is a generalization of traditional time-invariant (i.e., static) problems, and can naturally model: a) control problems where one seeks to generate a control action depending on a (parametric) varying optimization problem [2]–[4]; b) signal processing problems [5], where states of a dynamical process are estimated online based on time-varying observations – including time-varying compressive sensing settings [6]–[11]; and, c) inferential problems on dynamic networks [12]. Additional application domains include robotics [13]–[15], smart grids [16]–[18], economics [19], real-time

magnetic resonance imaging (MRI) [20], and human migration studies [21].

The problem (1) might be solved in a centralized setting based on a continuous time platform [22]–[27]; however, here we focus on a discrete time setting. The reason for this choice is motivated by the widespread use of digital computing units, such as control units (actuators) and digital sensors. In this context, we envision that our optimization problem will change in response to measurements taken at discrete time steps and its solution could provide control actions to be implemented on digital control units.

Therefore, we use sampling arguments to reinterpret (1) as a sequence of time-invariant problems. In particular, upon sampling the objective functions $f(\boldsymbol{x}; t)$ at time instants t_k, $k = 0, 1, 2, \ldots$, where the sampling period $h := t_k - t_{k-1}$ can be chosen arbitrarily small, one can solve the sequence of time-invariant problems

$$\boldsymbol{x}^*(t_k) := \operatorname*{argmin}_{\boldsymbol{x} \in X} f(\boldsymbol{x}; t_k), \quad k \in \mathbb{N}. \tag{2}$$

By decreasing h, an arbitrary accuracy may be achieved when approximating problem (1) with (2). However, solving (2) for each sampling time t_k may not be computationally affordable in many application domains, even for moderate-size problems[1].

Focusing on unconstrained optimization problems, the works in [29], [30] developed a prediction-correction method to find and track the solution trajectory $\boldsymbol{x}^*(t)$ up to a bounded asymptotical error, starting from an arbitrary guess $\boldsymbol{x}_0$. This methodology arises from non-stationary optimization [31], [32], parametric programming [19], [33]–[35], and continuation methods in numerical mathematics [36]. It also resembles evolutionary variational inequalities [21], [37] and path-following methods in interior point solvers [38]. This paper significantly broadens the method of [29] by offering the following contributions.

(i) We develop prediction-correction methods to track the solutions of the time-varying *constrained* problems (1);

(ii) We develop first-order algorithms that do not involve the computation of the inverse of the Hessian of the cost function, as required in [29]; the proposed prediction-correction method is computationally lighter, as it requires only matrix-vector multiplications. Further, we offer a trade-off between tracking capabilities and computational effort; and

(iii) We show that the proposed prediction-correction algorithm improves on the method in [29] when applied

[1]Note that here we focus on constant sampling periods; variable ones induced by self-triggering have been considered in unconstrained problems [28] and could be considered in constrained problems in future works.

to unconstrained optimization problems; particularly, it exhibits enhanced local convergence properties by relying on Newton-like prediction steps.

The design and analysis of proposed prediction-correction methods are grounded on the theory of generalized equations and implicit function theorems [34].

Organization. In Section II, we describe the prediction-correction methodology for constrained time-varying optimization problems. The special case of unconstrained case is developed in Section II-D. Convergence analysis is discussed in Section III, both for global $O(h)$ convergence and local $O(h^2)$ convergence. Numerical examples are displayed in Section IV – including a realistic application in real-time control of energy resources –, and in Section V, we draw our conclusions. The proofs of all the propositions and theorems are given in the appendices.

Notation. Vectors are written as $\boldsymbol{x} \in \mathbb{R}^n$ and matrices as $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. We use $\| \cdot \|$ to denote the Euclidean norm in the vector space, and the respective induced norms for matrices and tensors. The gradient of the function $f(\boldsymbol{x}; t)$ with respect to $\boldsymbol{x}$ at the point $(\boldsymbol{x}, t)$ is denoted as $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}; t) \in \mathbb{R}^n$, while the partial derivative of the same function with respect to (w.r.t.) t at $(\boldsymbol{x}, t)$ is written as $\nabla_t f(\boldsymbol{x}; t) \in \mathbb{R}$. Similarly, the notation $\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}; t) \in \mathbb{R}^{n \times n}$ denotes the Hessian of $f(\boldsymbol{x}; t)$ w.r.t. $\boldsymbol{x}$ at $(\boldsymbol{x}, t)$, whereas $\nabla_{t\boldsymbol{x}} f(\boldsymbol{x}; t) \in \mathbb{R}^n$ denotes the partial derivative of the gradient of $f(\boldsymbol{x}; t)$ w.r.t. the time t at $(\boldsymbol{x}, t)$, i.e. the mixed first-order partial derivative vector of the objective. The tensor $\nabla_{\boldsymbol{xxx}} f(\boldsymbol{x}; t) \in \mathbb{R}^{n \times n \times n}$ indicates the third derivative of $f(\boldsymbol{x}; t)$ w.r.t. $\boldsymbol{x}$ at $(\boldsymbol{x}, t)$, the matrix $\nabla_{\boldsymbol{x}t\boldsymbol{x}} f(\boldsymbol{x}; t) = \nabla_{t\boldsymbol{xx}} f(\boldsymbol{x}; t) \in \mathbb{R}^{n \times n}$ indicates the time derivative of the Hessian of $f(\boldsymbol{x}; t)$ w.r.t. the time t at $(\boldsymbol{x}, t)$, and the vector $\nabla_{tt\boldsymbol{x}} f(\boldsymbol{x}; t) \in \mathbb{R}^n$ indicates the second derivative in time of the gradient of $f(\boldsymbol{x}; t)$ w.r.t. the time t at $(\boldsymbol{x}, t)$. The indicator function is indicated as $\iota_X(\boldsymbol{x})$ for the convex set $X \subseteq \mathbb{R}^n$; by definition $\iota_X(\boldsymbol{x}) = 0$ when $\boldsymbol{x} \in X$ and $+\infty$ otherwise. The subdifferential of $\iota_{\mathbf{X}}$ is the set-valued map known as normal cone $N_X : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ (Cf. [34]).

II. PREDICTION-CORRECTION STRATEGY

In this section, we first focus on problem (1) and design a prediction-correction algorithm to track the (unique) trajectory of the optimal solution. As explained in the introduction, consider sampling (1) at times t_k, $k \in \mathbb{N}$, and constructing a sequence of time-invariant problems (2). In lieu of solving (2) at each time step, the goal of the prediction-correction strategy is to determine an approximate optimizer for (1) at t_{k+1} in a computationally affordable way, given the current approximate optimizer at t_k.

A. Prediction

Suppose that $\boldsymbol{x}_k$ is an approximate solution of (2) at time t_k. Given $\boldsymbol{x}_k$, the prediction step seeks an approximate optimizer for (1) at t_{k+1}, given the only information available at time t_k. Let $\boldsymbol{x}_{k+1|k}$ denote the output of the prediction step, which is computed as explained next.

Notice first that solving the time-invariant problem (2) associated with time t_k is equivalent to solving the generalized equation

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_k) + N_X(\boldsymbol{x}^*(t_k)) \ni \mathbf{0} \quad (3)$$

where N_X is the normal cone operator, while $\boldsymbol{x}^*(t_k)$ is the optimizer of (2) at t_k. Although many equivalent ways to formulate the solution of (2) exist, e.g., as the fixed-point of a properly defined operator, here we focus on generalized equations to build on the implicit function theorems of [19], [34].

With (3) in place, the prediction step seeks the solution of the following perturbed generalized equation

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_{k+1|k}; t_{k+1}) + N_X(\boldsymbol{x}_{k+1|k}) \approx \\ \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k; t_k) + \nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k)(\boldsymbol{x}_{k+1|k} - \boldsymbol{x}_k) \\ + h \nabla_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k) + N_X(\boldsymbol{x}_{k+1|k}) \ni \mathbf{0}. \quad (4)$$

That is, the prediction step produces a solution that is optimal w.r.t. a perturbed (first-order) version of the original generalized equation (3). The idea of this perturbed approximate version is motivated by the fact that one would like to solve $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_{k+1|k}; t_{k+1}) + N_X(\boldsymbol{x}_{k+1|k}) \ni \mathbf{0}$, but that is not possible at time t_k. Therefore, we perturb said generalized equation by a backward Taylor expansion, in a way that is possible to solve it at time t_k.

We can now replace (4) with the following equivalent formulation

$$\boldsymbol{x}_{k+1|k} = \underset{\boldsymbol{x} \in X}{\operatorname{argmin}} \left\{ \frac{1}{2} \boldsymbol{x}^\mathsf{T} \nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k) \boldsymbol{x} \right. \\ \left. + (\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k; t_k) + h \nabla_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k) - \nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k) \boldsymbol{x}_k)^\mathsf{T} \boldsymbol{x} \right\}, \quad (5)$$

where we can notice that the normal cone operator yields the feasible set over which the optimization is computed.

In lieu of seeking an exact solution of (5) (which is a constrained optimization problem with quadratic cost), consider the less computational demanding task of finding an approximate solution of (5) by computing a number of projected gradient descent steps – the first key step towards a *first-order* prediction-correction method (Note that we consciously avoid Newton's method here for the high computational complexity burden of computing the Hessian inverse). Particularly, let $\widehat{\boldsymbol{x}}^0$ be a dummy variable initialized as $\widehat{\boldsymbol{x}}^0 = \boldsymbol{x}_k$; then, the following steps are performed:

$$\widehat{\boldsymbol{x}}^{p+1} = \mathbb{P}_X[\widehat{\boldsymbol{x}}^p - \alpha(\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k)(\widehat{\boldsymbol{x}}^p - \boldsymbol{x}_k) \\ + h \nabla_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k) + \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k; t_k))], \quad (6)$$

for $p = 0, 1, \ldots, P-1$, where P is a pre-determined number of gradient steps, $\alpha > 0$ is the stepsize, and $\mathbb{P}_X$ is the projection operator over the convex set X. Once P steps are performed, $\tilde{\boldsymbol{x}}_{k+1|k}$ is set to:

$$\tilde{\boldsymbol{x}}_{k+1|k} = \widehat{\boldsymbol{x}}^P. \quad (7)$$

B. Correction

Once the cost function $f(\cdot; t_{k+1})$ becomes available, the correction step is performed to refine the estimate of the optimal solution $\boldsymbol{x}^*(t_{k+1})$. To this end, a first-order projected gradient method is considered next. Particularly, let $\widehat{\boldsymbol{x}}^0 =$

Algorithm 1 Constrained First-Order Prediction-Correction (C-FOPC)

Require: Initial variable $\boldsymbol{x}_0$. Initial objective function $f(\boldsymbol{x}; t_0)$, no. of prediction steps P and correction steps C

1: **for** $k = 0, 1, 2, \ldots$ **do**
2:    // time t_k
3:    Prediction: initialize $\widehat{\boldsymbol{x}}^0 = \boldsymbol{x}_k$
4:    **for** $p = 0 : P - 1$ **do**
5:       Predict the variable by the gradient step [cf (6)]

$$\widehat{\boldsymbol{x}}^{p+1} = \mathbb{P}_X[\widehat{\boldsymbol{x}}^p - \alpha(\nabla_{\boldsymbol{xx}}f(\boldsymbol{x}_k; t_k)(\widehat{\boldsymbol{x}}^p - \boldsymbol{x}_k) + \\ h\,\nabla_{t\boldsymbol{x}}f(\boldsymbol{x}_k; t_k) + \nabla_{\boldsymbol{x}}f(\boldsymbol{x}_k; t_k))]$$

6:    **end for**
7:    Set the predicted variable $\tilde{\boldsymbol{x}}_{k+1|k} = \widehat{\boldsymbol{x}}^P$
8:    // time t_{k+1}
9:    Acquire the updated function $f(\boldsymbol{x}; t_{k+1})$
10:   Initialize the sequence of corrected variables $\widehat{\boldsymbol{x}}^0 = \tilde{\boldsymbol{x}}_{k+1|k}$
11:   **for** $c = 0 : C - 1$ **do**
12:      Correct the variable by the gradient step [cf (8)]

$$\widehat{\boldsymbol{x}}^{c+1} = \mathbb{P}_X[\widehat{\boldsymbol{x}}^c - \beta(\nabla_{\boldsymbol{x}}f(\widehat{\boldsymbol{x}}^c; t_{k+1})]$$

13:   **end for**
14:   Set the corrected variable $\boldsymbol{x}_{k+1} = \widehat{\boldsymbol{x}}^C$
15: **end for**

$\tilde{\boldsymbol{x}}_{k+1|k}$ be a dummy variable; then, consider the following projected gradient steps

$$\widehat{\boldsymbol{x}}^{c+1} = \mathbb{P}_X[\widehat{\boldsymbol{x}}^c - \beta(\nabla_{\boldsymbol{x}}f(\widehat{\boldsymbol{x}}^c; t_{k+1})], \tag{8}$$

for $c = 0, 1, \ldots, C - 1$, where C is a predetermined number of gradient steps and $\beta > 0$ the stepsize. The estimate of the optimal solution $\boldsymbol{x}_{k+1}^*$ is then computed as $\boldsymbol{x}_{k+1} = \widehat{\boldsymbol{x}}^C$.

C. Complete Algorithm

The complete algorithm Constrained - First Order Prediction Correction (C-FOPC) is tabulated as Algorithm 1. Steps 4-7 are utilized to compute $\tilde{\boldsymbol{x}}_{k+1|k}$ based on the information available at t_k. Provided that the projection operator is easy to carry out (set X is simple, e.g., it is an ℓ_2 or ℓ_∞ norm), and the Hessian is easy to evaluate, the computational complexity of these steps is $O(Pn^2)$, which is quadratic (due to matrix-vector multiplications) in the number of scalar decision variables. This is in contrast with the algorithms presented in [29], which involve the computation of the Hessian inverse and whose computational complexity is therefore $O(n^3)$. Steps 10-14 are utilized to compute $\boldsymbol{x}_{k+1}$, based on the information available at t_{k+1}. Provided that the projection operator is easy to perform (set X is simple) and the gradient is easy to evaluate, the computational complexity of these steps is $O(Cn)$, which is linear in the number of scalar decision variables.

Remark 1: [Time derivative approximation] The time derivative of the gradient $\nabla_{t\boldsymbol{x}}f(\boldsymbol{x}; t)$ can be substituted with an approximate version, as explained in [29].

D. Special Case: Unconstrained Problems

In this section, we focus on the special case of unconstrained problems (i.e., $X = \mathbb{R}^n$). Although [29] has given an extensive characterization of methods for unconstrained problems, we

will see in this section that further important improvements can be achieved based on the prediction generalized equation (4).

For unconstrained problems, the prediction equation (4) can be rewritten as

$$\nabla_{\boldsymbol{x}}f(\boldsymbol{x}_{k+1|k}; t_{k+1}) \approx \nabla_{\boldsymbol{x}}f(\boldsymbol{x}_k; t_k) + \\ \nabla_{\boldsymbol{xx}}f(\boldsymbol{x}_k; t_k)(\boldsymbol{x}_{k+1|k} - \boldsymbol{x}_k) + h\,\nabla_{t\boldsymbol{x}}f(\boldsymbol{x}_k; t_k) = \boldsymbol{0}. \tag{9}$$

Since the suboptimality at time t_k can be easily characterized by the gradient, i.e., $\nabla_{\boldsymbol{x}}f(\boldsymbol{x}_k; t_k)$, we can also modify (9) to tune the prediction step. A way to do that is to require $\boldsymbol{x}_{k+1|k}$ to reduce the suboptimality by a factor of $1 - \gamma \in [0, 1]$, and pose the problem as

$$\nabla_{\boldsymbol{x}}f(\boldsymbol{x}_{k+1|k}; t_{k+1}) \approx \nabla_{\boldsymbol{x}}f(\boldsymbol{x}_k; t_k) + \nabla_{\boldsymbol{xx}}f(\boldsymbol{x}_k; t_k) \times \\ (\boldsymbol{x}_{k+1|k} - \boldsymbol{x}_k) + h\,\nabla_{t\boldsymbol{x}}f(\boldsymbol{x}_k; t_k) = (1 - \gamma)\nabla_{\boldsymbol{x}}f(\boldsymbol{x}_k; t_k). \tag{10}$$

When $\gamma = 1$, (10) boils down to (9) (when one seeks an "optimal prediction"); on the other hand, when $\gamma = 0$, (10) coincide with [29], where a prediction vector that maintains the same suboptimality (i.e., the same gradient) between successive time steps is sought. Notice that the possibility of tuning the algorithm via γ is possible only for unconstrained optimization problems, since we have access to a "suboptimality measure".

From (10) we obtain the prediction vector via the following update

$$\boldsymbol{x}_{k+1|k} = \boldsymbol{x}_k - \nabla_{\boldsymbol{xx}}f(\boldsymbol{x}_k; t_k)^{-1} \times \\ (h\,\nabla_{t\boldsymbol{x}}f(\boldsymbol{x}_k; t_k) + \gamma\,\nabla_{\boldsymbol{x}}f(\boldsymbol{x}_k; t_k)), \tag{11}$$

which combines a Newton-like step in the direction on the changing cost function, and a (damped) Newton's step towards the optimizer at time t_k. When $\gamma = 0$, (11) coincides with the prediction step in [29]. To obtain a first-order update, it is then easy to modify (6) as

$$\widehat{\boldsymbol{x}}^{p+1} = \widehat{\boldsymbol{x}}^p - \alpha(\nabla_{\boldsymbol{xx}}f(\boldsymbol{x}_k; t_k)(\widehat{\boldsymbol{x}}^p - \boldsymbol{x}_k) \\ + h\,\nabla_{t\boldsymbol{x}}f(\boldsymbol{x}_k; t_k) + \gamma\,\nabla_{\boldsymbol{x}}f(\boldsymbol{x}_k; t_k)), \tag{12}$$

where the notation is the same as in (6). On the other hand, the correction update becomes in this case

$$\widehat{\boldsymbol{x}}^{c+1} = \widehat{\boldsymbol{x}}^c - \beta(\nabla_{\boldsymbol{x}}f(\widehat{\boldsymbol{x}}^c; t_{k+1}). \tag{13}$$

The resultant Unconstrained - First Order Prediction Correction (U-FOPC) method is tabulated as Algorithm 2.

III. CONVERGENCE ANALYSIS

In this section, we establish analytical results to bound the discrepancy between the optimal solution $\boldsymbol{x}^*(t)$ and the iterates $\boldsymbol{x}_k$ produced by the prediction-correction schemes developed in Section II. Particularly, we will show that $\boldsymbol{x}_k$ tracks $\boldsymbol{x}^*(t)$ up to an error term that depends on the discrete-time sampling period. To this end, some technical conditions are required as stated next

Assumption 1: **The function $f(\boldsymbol{x}; t)$ is twice differentiable and m-strongly convex in $\boldsymbol{x} \in X$ and uniformly**

Algorithm 2 Unconstrained First-Order Prediction-Correction (U-FOPC)

Require: Initial variable $\boldsymbol{x}_0$. Initial objective function $f(\boldsymbol{x}; t_0)$, no. of prediction steps $P+1$ and correction steps $C+1$, sub-optimality requirement $\gamma \in [0,1]$

1: **for** $k = 0, 1, 2, \ldots$ **do**
2:    // time t_k
3:    Prediction: initialize $\widehat{\boldsymbol{x}}^0 = \boldsymbol{x}_k$
4:    **for** $p = 0 : P - 1$ **do**
5:       Predict the variable by the gradient step [cf (6)]

$$\widehat{\boldsymbol{x}}^{p+1} = \widehat{\boldsymbol{x}}^p - \alpha(\nabla_{\boldsymbol{x}\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)(\widehat{\boldsymbol{x}}^p - \boldsymbol{x}_k)$$
$$+ h\, \nabla_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k) + \gamma\, \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k; t_k))$$

6:    **end for**
7:    Set the predicted variable $\tilde{\boldsymbol{x}}_{k+1|k} = \widehat{\boldsymbol{x}}^P$
8:    // time t_{k+1}
9:    Acquire the updated function $f(\boldsymbol{x}; t_{k+1})$
10:    Initialize the sequence of corrected variables $\widehat{\boldsymbol{x}}^0 = \tilde{\boldsymbol{x}}_{k+1|k}$
11:    **for** $c = 0 : C - 1$ **do**
12:       Correct the variable by the gradient step [cf (8)]

$$\widehat{\boldsymbol{x}}^{c+1} = \mathbb{P}_X[\widehat{\boldsymbol{x}}^c - \beta(\nabla_{\boldsymbol{x}} f(\widehat{\boldsymbol{x}}^c; t_{k+1})]$$

13:    **end for**
14:    Set the corrected variable $\boldsymbol{x}_{k+1} = \widehat{\boldsymbol{x}}^C$
15: **end for**

---

in t; that is, the Hessian of $f(\boldsymbol{x}; t)$ with respect to $\boldsymbol{x}$ is bounded below by m for each $\boldsymbol{x} \in X$ and uniformly in t,

$$\nabla_{\boldsymbol{x}\boldsymbol{x}} f(\boldsymbol{x}; t) \succeq m\mathbf{I}, \quad \forall \boldsymbol{x} \in X, t.$$

In addition, the function $f(\boldsymbol{x}; t)$ has bounded second and third order derivatives with respect to $\boldsymbol{x} \in X$ and t:

$$\|\nabla_{\boldsymbol{x}\boldsymbol{x}} f(\boldsymbol{x}; t)\| \leqslant L, \|\nabla_{t\boldsymbol{x}} f(\boldsymbol{x}; t)\| \leqslant C_0.$$

Assumption 2: **The function $f(\boldsymbol{x}; t)$ has bounded third order derivatives with respect to $\boldsymbol{x} \in X$ and t:**

$$\|\nabla_{\boldsymbol{x}\boldsymbol{x}\boldsymbol{x}} f(x; t)\| \leqslant C_1, \|\nabla_{\boldsymbol{x}t\boldsymbol{x}} f(\boldsymbol{x}; t)\| \leqslant C_2, \|\nabla_{tt\boldsymbol{x}} f(\boldsymbol{x}; t)\| \leqslant C_3.$$

Assumption 1 guarantees that problem (1) is strongly convex and has a *unique* solution for each time instance. Uniqueness of the solution implies that the solution trajectory is also unique. Assumption1 also ensures that the Hessian is bounded from above; this property is equivalent to the Lipschitz continuity of the gradient. This setting is common in the the time-varying optimization domain; see, for instance [5], [19], [29], [30], [32], [39]. Assumption 2 ensures that the third derivative tensor $\nabla_{\boldsymbol{x}\boldsymbol{x}\boldsymbol{x}} f(x; t)$ is bounded above (typically required for the analysis of Newton-type algorithms). Assumptions 1-2 impose boundedness of the temporal variability of gradient and Hessian. These last properties ensure the possibility to build a prediction scheme based on the knowledge of (or an estimate of) how the function and its derivatives change over time. A similar assumption was required (albeit only locally) for the local convergence analysis in [19, Eq. (3.2)].

Assumption 1 is sufficient to show that the solution *mapping* $t \mapsto \boldsymbol{x}^*(t)$ is single-valued and locally Lipschitz continuous in t; in particular, from [34, Theorem 2F.10] we have that

$$\|\boldsymbol{x}^*(t_{k+1}) - \boldsymbol{x}^*(t_k)\| \leqslant \frac{1}{m} \|\nabla_{t\boldsymbol{x}} f(\boldsymbol{x}; t)\|(t_{k+1} - t_k) \leqslant \frac{C_0 h}{m}, \tag{14}$$

for sufficiently small sampling periods h. This result established link between the sampling period h and the temporal variability of the optimal solutions; further, (14) will be utilized to substantiate convergence and tracking capabilities of the proposed prediction-correction methods.

A. Constrained algorithm global $O(h)$ convergence

We study the convergence properties of the sequence $\{\boldsymbol{x}_k\}_{k \in \mathbb{N}}$ generated by the algorithm C-FOPC, for different choices of the stepsize. In the following theorem, we show that the optimality gap $\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|$ converges exponentially to a given error bound, globally, i.e., from an arbitrary $\boldsymbol{x}_0$.

Theorem 1: **Consider the sequence $\{\boldsymbol{x}_k\}_{k \in \mathbb{N}}$ generated by the C-FOPC method, and let Assumption 1 hold true. Let $\boldsymbol{x}^*(t_k)$ be the optimizer of (1) at time t_k. Define the following quantities**

$$\varrho_P = \max\{|1 - \alpha m|, |1 - \alpha L|\}, \; \varrho_C = \max\{|1 - \beta m|, |1 - \beta L|\}, \tag{15}$$

and choose the stepsizes α and β as

$$\alpha < 2/L, \quad \beta < 2/L. \tag{16}$$

Further, select the number of correction steps C in a way that

$$\tau_0 := \varrho_C^C \left[\varrho_P^P + (\varrho_P^P + 1)\frac{2L}{m} \right] < 1. \tag{17}$$

Then, the sequence $\{\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|\}_{k \in \mathbb{N}}$ converges linearly with rate τ_0 to an asymptotical error bound, and

$$\limsup_{k \to \infty} \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| = O(\varrho_C^C h). \tag{18}$$

Proof: See Appendix B, where we also derive all the constants in the right-hand side of (18). ∎

Theorem 1 says that the sequence $\{\boldsymbol{x}_k\}_{k \in \mathbb{N}}$ generated by C-FOPC converges to a neighborhood of the optimal solution trajectory $\boldsymbol{x}^*(t)$. In particular, for a choice of prediction and correction steps P and C, the sequence converge linearly to the asymptotic bound. The bound depends linearly on the sampling period and exponentially on the correction steps C. When one performs the correction step exactly, i.e., $C \to \infty$, then the asymptotic bound goes to zero (in fact, in that case each of the time-invariant problems is solved exactly). Finally, if one looks at the condition (17), one discovers that for global convergence one needs more that standard $\varrho_C^C < 1$: in particular C has to be chosen high enough to counteract any errors that come from the prediction step. We see next that this condition can be weakened in the case in which the cost function has higher order smoothness properties like the ones required by Assumption 2.

B. Constrained algorithm local $O(h^2)$ convergence

We study now the local convergence properties of the sequence $\{\boldsymbol{x}_k\}_{k \in \mathbb{N}}$ generated by the algorithm C-FOPC under higher order smoothness (Assumption 2), and we show that starting from a small enough optimality gap $\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|$, and for a small enough sampling period h, the error bound can be reduced from $O(h)$ to $O(h^2)$.

Theorem 2: **Consider the sequence $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ generated by the C-FOPC method, and let Assumptions 1-2 hold true. Let $\boldsymbol{x}^*(t_k)$ be the optimizer of (1) at time t_k. Define the following quantities**

$$\varrho_P = \max\{|1-\alpha m|, |1-\alpha L|\}, \ \varrho_C = \max\{|1-\beta m|, |1-\beta L|\},$$
$$(19)$$

and choose the stepsizes α and β as

$$\alpha < 2/L, \quad \beta < 2/L. \qquad (20)$$

Further, select $\tau \in (0,1)$, the number of prediction steps P, and the number of correction steps C in a way that $\varrho_P^P \varrho_C^C < \tau$.

There exist an upper bound on the sampling period $\bar{h}$ and a convergence region $\bar{R}$, such that if the sampling period is chosen as $h \leqslant \bar{h}$ and the initial optimality gap satisfy $\|\boldsymbol{x}_0 - \boldsymbol{x}^*(t_0)\| \leqslant \bar{R}$, then the sequence $\{\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|\}_{k\in\mathbb{N}}$ converges linearly with rate τ to an asymptotical error bound, and

$$\limsup_{k\to\infty} \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| = O(h^2 \varrho_C^C) + O(h \varrho_P^P \varrho_C^C). \qquad (21)$$

In addition, the bounds $\bar{h}$ and $\bar{R}$ are given as

$$\bar{h} = \frac{\tau - \varrho_C^C \varrho_P^P}{\varrho_C^C(\varrho_P^P + 1)}\left(\frac{C_1 C_0}{m^2} + \frac{C_2}{m}\right)^{-1}, \bar{R} = \frac{2\,m}{C_1}\left(\frac{C_1 C_0}{m^2} + \frac{C_2}{m}\right)(\bar{h}-h).$$
$$(22)$$

Proof: See Appendix B, where we also derive all the constants in the right-hand side of (21). ∎

Theorem 2 asserts that the sequence $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ generated by C-FOPC locks to a neighborhood of the optimal solution trajectory $\boldsymbol{x}^*(t)$. In particular, for a choice of prediction and correction steps P and C, there exist an upper bound on the sampling period and an attraction region, such that if the sampling period is smaller than the bound and the initial optimality gap is in the attraction region, then the sequence converge (at least) linearly to an asymptotic bound. The bound depends on the sampling period and on the selection of prediction and correction steps P and C. When one performs an optimal prediction (that is $P \to \infty$), then the bound goes as $O(h^2)$, which is similar to the bounds derived in [29]. When one performs the correction step exactly, i.e., $C \to \infty$, then the asymptotic bound goes to zero (in fact, in that case each time-invariant problem is solved exactly).

The presence of an attraction region is due to mimicking a Newton's step in the prediction stage (that is, the presence of the gradient $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)$ in the generalized equation (4)). To further understand this point, consider the unconstrained case: as expressed in Section II-D, the prediction step optimization problem (5) has a closed-form solution as Eq. (11). The presence of the gradient $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)$ translates into the presence of the Hessian inverse in (11), which is a Newton's step. Our prediction strategy, even in the constrained case, mimics such step and inherits its pros and cons, e.g., the presence of a local convergence region. In addition, as in a Newton's method, when the function is quadratic, then $C_1 = 0$, and the convergence is global.

By putting together the results of Theorems 1-2, one can arrive at simple strategies for parameter selection for the designer. In particular, for convergence, the designer needs only to know the strong convexity constant m and the Lipschitz constant L, which is a standard requirement. Then, they can design the stepsizes α and β as in (15) (or equivalently (19)) and the number of prediction and correction steps as in (17). With this in place, in the worst case they obtain a $O(\varrho_C^C h)$ error bound, and in the best case – if the function satisfies Assumption 2, the optimality gap enters in the attraction region, and h is small enough – they obtain a better $O(h^2 \varrho_C^C) + O(h \varrho_P^P \varrho_C^C)$ error bound. This is the case, since the conditions on C and P are more stringent under Assumption 1 alone than with also the use of Assumption 2. This eliminates de-facto the requirement of knowing (or estimating) the bounds C_0, C_1, C_2, C_3, which may be difficult to do in practice.

C. Unconstrained algorithm convergence

The results of Theorems 1-2 can be tuned to the case of unconstrained problems as shown in the following.

Theorem 3: **Consider the sequence $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ generated by the U-FOPC method, and let Assumption 1 hold true. Let $\boldsymbol{x}^*(t_k)$ be the optimizer of (1) at time t_k for $X = \mathbb{R}^n$. Define the following quantities**

$$\varrho_P = \max\{|1-\alpha m|, |1-\alpha L|\}, \ \varrho_C = \max\{|1-\beta m|, |1-\beta L|\},$$
$$(23)$$

and choose stepsizes α and β as

$$\alpha < 2/L, \quad \beta < 2/L. \qquad (24)$$

Further, select the number of correction steps C in a way that

$$\tau_0 := \varrho_C^C\left[\varrho_P^P + (\varrho_P^P + 1)\left(1 - \gamma + \gamma\frac{2L}{m}\right)\right] < 1. \qquad (25)$$

Then, the sequence $\{\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|\}_{k\in\mathbb{N}}$ converges linearly with rate τ_0 to an asymptotical error bound, and

$$\limsup_{k\to\infty} \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| = O(\varrho_C^C h). \qquad (26)$$

Proof: See Appendix C, where we also derive all the constants in the right-hand side of (26). The proofs leverage generalized equation theory and the special unconstrained nature of the problem. ∎

Similarly to Theorem 1, Theorem 1 is a global $O(h)$ convergence result. From condition (25), it would seem that the best choice of γ would be $\gamma = 0$; however this may not be the case when Assumption 2 holds.

Theorem 4: **Consider the sequence $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ generated by the U-FOPC method. Let Assumptions 1-2 hold true. Let $\boldsymbol{x}^*(t_k)$ be the optimizer of (1) at time t_k for $X = \mathbb{R}^n$. Define,**

$$\varrho_P = \max\{|1-\alpha m|, |1-\alpha L|\}, \ \varrho_C = \max\{|1-\beta m|, |1-\beta L|\},$$
$$(27)$$

and set the stepsizes α and β as

$$\alpha < 2/L, \quad \beta < 2/L. \qquad (28)$$

Let $\tau \in (0,1)$, P, C be such that

$$(1-\gamma)\varrho_C^C(1+\varrho_P^P) + \varrho_P^P\varrho_C^C < \tau. \tag{29}$$

There exist an upper bound on the sampling period $\bar{h}$ and a convergence region $\bar{R}$, such that if the sampling period is chosen as $h \leqslant \bar{h}$ and the initial optimality gap satisfy $\|x_0 - x^*(t_0)\| \leqslant \bar{R}$, then the sequence $\{\|x_k - x^*(t_k)\|\}_{k\in\mathbb{N}}$ converges linearly with rate τ to an asymptotical error bound, and

$$\limsup_{k\to\infty} \|x_k - x^*(t_k)\| = O(h^2\,\varrho_C^C) + O(h\,\varrho_P^P\varrho_C^C). \tag{30}$$

In addition, the bounds $\bar{h}$ and $\bar{R}$ are given as

$$\bar{h} = \left(\frac{\tau - \varrho_C^C\varrho_P^P}{\varrho_C^C(\varrho_P^P+1)} - 1 + \gamma\right)\left(\frac{C_1C_0}{m^2} + \frac{C_2}{m}\right)^{-1}, \tag{31}$$

$$\bar{R} = \frac{2\,m}{\gamma C_1}\left(\frac{C_1C_0}{m^2} + \frac{C_2}{m}\right)(\bar{h} - h). \tag{32}$$

Proof: See Appendix C, where we also derive all the constants in the right-hand side of (30). The proofs leverage generalized equation theory and the special unconstrained nature of the problem. ∎

Theorem 4 is a local $O(h^2)$ convergence result. It can be seen as a generalization of Theorem 1 in [29] and a special, yet more detailed, version of Theorem 2. In particular, by properly selecting the parameter γ, one obtains the results of both Theorem 2 and [29]; when $\gamma = 1$, we obtain the results of Theorem 2 and this indicates that considering constrained problems does not add extra errors to the asymptotical bounds. On the other hand, if $\gamma = 0$, we are implicitly assuming that the prediction stage leads one to navigate into the iso-residual manifold; since $\bar{R} \to \infty$, we also obtain global convergence. Furthermore, one can use γ as a tuning mechanism to enlarge the convergence region.

Finally, for $\gamma = 0$ and $P \to \infty$, then the results of theorem (30) boil down to those in [29]. In particular, from (31), it can be seen that the sampling period must satisfy the following relationship

$$\varrho_C^C\left[1 + h\left(\frac{C_1C_0}{m^2} + \frac{C_2}{m}\right)\right] < \tau < 1, \tag{33}$$

which is the same requirement of [29, Theorem 1].

IV. NUMERICAL EXPERIMENTS

In this section, we report three numerical examples to showcase the performance of the proposed algorithms. First, we analyze the unconstrained case (Algorithm 2), then we switch our attention to the constrained one (Algorithm 1). Finally, we present a realistic simulation stemming from real-time control of energy resources.

A. Unconstrained example

We use the same scalar example of [29, Section IV.A], and in particular we consider a time-varying cost function of the form

$$\min_{x\in\mathbb{R}} f(x;t) := \frac{1}{2}(x - \cos(\omega t))^2 + \kappa\log[1 + \exp(\mu x)]. \tag{34}$$

Fig. 1. Error with respect to the sampling time t_k for different algorithms applied to the scalar problem (34), with $h = 0.1$.

Fig. 2. Asymptotical worst case error floor with respect to the sampling time interval h for different algorithms applied to the scalar problem (34).

The function in (34) represents, for instance, the goal of staying close to a periodically varying trajectory plus a logistic term that penalizes large values of x. The terms ω, κ, and μ are arbitrary nonnegative scalar parameters. In our experiments these parameters are set to $\omega = \pi/2$, $\kappa = 2$, and $\mu = 1.75$. The function $f(x;t)$ satisfies all the conditions in Assumptions 1 and 2. In particular, $m = 1$ and $L = 2.53$.

We choose the constant stepsizes as $\alpha = \beta = 0.56 < 2/L$ in the gradient methods stated in Algorithm 2 and initialize $x_0 = 0$ for all the algorithms. According to Theorem 3, for $\gamma = 1$, we need $C > 2$ to ensure $\tau_0 < 1$ and therefore $O(h)$ convergence. In the case of Theorem 4, for $\gamma = 0$, $P = 1$ and $C = 3$: we can set $\tau > 0.16$, the sampling period needs to be chosen as $h \leqslant \bar{h} = 4.2$, and the convergence region is the whole $\mathbb{R}$, while for $\gamma = 1$, $P = 1$ and $C = 3$: we can set $\tau > 0.038$, $\bar{h} = 4.84$, and $\bar{R} = 12.7$ (notice that for greater P or C the requirements are less stringent).

In Figure 1, we plot the error $\|x_k - x^*(t_k)\|$ versus the

discrete time t_k for a sampling period of $h = 0.1$, for different schemes (we fix the correction steps at $C = 3$ for all the algorithms and we study the behavior for different prediction steps P and γ values). Observe that the running gradient method [32], that is U-FOPC with no prediction and one correction step performs the worst, while U-FOPC with exact prediction (that is $P = \infty$) performs the best[2]. Furthermore, we can notice that increasing P, one obtains better and better asymptotical error, with the drawback of an increased computational burden. Lastly, we notice how $\gamma = 1$, that is having a Newton-like prediction step helps in achieving a faster convergence, and yet it appears that the asymptotical error is slightly greater than using $\gamma = 0$ (i.e., tangential prediction).

The differences in performance can be also appreciated by varying h and observing the worst case error floor size which is defined as $\max_{k > \bar{k}}\{\|x_k - x^*(t_k)\|\}$, where $\bar{k} = 10^4$ in the simulations. Figure 2 illustrates the error as a function of h. The performance differences between the proposed methods that may be observed here corroborate the differences seen in Figure 1. In particular, the U-FOPC method with $P = 0, C = 1$ achieves the largest worst case error bound, while exact prediction with $\gamma = 0$ attains the best case error bound. Notice also the dashed lines displaying the theoretical performance of $O(h)$, $O(h^2)$, and the fact that for any finite P there is a trade-off between computational complexity and asymptotical error[3].

B. Constrained example

We consider here a mid-size optimization problem consisting of $n = 1000$ scalar variables. The cost function we study has the form

$$f(\boldsymbol{x};t) = \frac{1}{2}\|\boldsymbol{x}+\mathbf{1}_n\|_{\mathbf{Q}}^2 + \sum_{i=1}^n \kappa_i \sin^2(\omega t + \varphi_i)\,\exp(\mu(x_{(i)}-2)^2),$$

$$(35)$$

where we have defined $\mathbf{1}_n$ as the column vector of all ones of dimension n, while $x_{(i)}$ is the i-th component of $\boldsymbol{x} \in \mathbb{R}^n$. In addition, the matrix $\mathbf{Q}$ is chosen as $\mathbf{Q} = \mathbf{I}_n + \boldsymbol{\mu}\boldsymbol{\mu}^\mathsf{T}/n$ with $\boldsymbol{\mu}$ being a vector randomly generated by a normal distribution of mean 0 and variance 1, $\kappa_i \sim \mathcal{U}_{[0,1]}$, $\omega = 0.1\pi$, $\varphi \sim \mathcal{N}(0,\pi)$, and $\mu = 0.25$.

We study the time-varying problem

$$\underset{\boldsymbol{x} \in [0,0.4]^n}{\text{minimize}}\ f(\boldsymbol{x};t).\qquad(36)$$

We notice that the cost function f verifies the Assumptions 1-2 on on $[0,0.4]^n$, which is our optimization set (even though it would not satisfy them on the whole $\mathbb{R}^n$). In particular, $m = 1$ and $L = 6.07$.

One could run a similar analysis as the unconstrained example, however here we focus on realistic run-time constraints. Every time a new function is available, a number of correction steps are performed. The number depends on

how fast we need the corrected variable to be available and the computational time necessary to compute the gradient and perform the correction step. We fix at $r_1 h$, with $r_1 < 1$ the time allocated for the correction steps, while t_C is the time to perform one correction step. For the above considerations, we can afford to run

$$C = \lfloor r_1 h / t_\mathrm{C} \rfloor,\qquad(37)$$

correction steps. After the corrected variable is available, one can use it for the decision making process (which may require extra time to be performed). For the time-varying algorithm perspective, one can use the variable to either run P gradient prediction, or C' extra correction steps (to improve the corrected variable for having a better starting point when a new function becomes available). Fix at $r_2 h$, with $r_2 < 1$ the time allocated for the prediction (or extra correction) steps. The affordable number of prediction steps can be determined considering that P prediction steps require a time equal to $\bar{t} + P t_\mathrm{P}$, where $\bar{t}$ is the time required to evaluate the Hessian, gradient, and time derivative of the gradient, while t_P is the time to perform one prediction calculation. Thus,

$$P = \lfloor (r_2 h - \bar{t})/t_\mathrm{P} \rfloor.\qquad(38)$$

The affordable extra correction steps C' can be computed as in (37), substituting r_1 with r_2.

In the simulation example, we choose $r_1 = r_2 = 0.5$, while by running the experiments on a 1.8 GHz Intel Core i5, we empirically fix $t_\mathrm{C} = .76$ ms, $\bar{t} = 10$ ms, $t_\mathrm{P} = .62$ ms. Note that the time that would be needed to solve the prediction step exactly (by solving a quadratic program) is 190 ms, which is not affordable in the considered sampling period range.

In addition, we consider the situation in which one can use the whole sampling period to do correction, that is $r_1 = 1$, while $r_2 = 0$, and we call this case *total correction*. This situation is particularly interesting when one has to make a choice whether to stop the correction steps to perform prediction, or to continue to do correction steps till a new function evaluation becomes available. Note that the correction+extra correction strategy is different from the total correction one, since the error is computed with the corrected variable (which is used for the decision making process), that is after $r_1 h$.

In Figure 3, we report the asymptotical worst-case error w.r.t the sampling period for the three considered cases (correction+extra correction, total correction, and prediction-correction, i.e. the C-FOPC algorithm)[4], while the number of prediction steps and correction steps are optimized via the available resources as in Eq.s (37)-(38). With the simulation parameters, for $h = 6$ ms, we can perform $C = C' = 3$ steps of correction and extra correction, or $C = 7$ steps of (total) correction. For $h = 40$ ms, these values are $C = C' = 26$ and $C = 52$, respectively. For the prediction-correction strategy, for $h = 22$ ms, then $C = 14$ and $P = 1$, while for $h = 40$ ms, $C = 26$ and $P = 16$.

For sampling times below 22 ms, prediction cannot be performed due to time constraints. For sampling periods greater or equal than 22 ms, prediction can be performed and for

---

[2]The running gradient is a standard gradient descent with constant stepsize, where at each iteration one performs C correction steps and the cost changes at each sampling time.

[3]The simulation code is made available online to further analyze how different parameters can affect the algorithm trade-offs.

[4]Note that both the correction+extra correction and the total correction strategies are in fact (running) projected gradient descent algorithms.

Fig. 3. Asymptotical worst case error floor with respect to the sampling time interval h for different algorithms applied to (36)

Fig. 4. Trajectories of $\{\boldsymbol{x}_k\}$ and optimal trajectory $\boldsymbol{x}^*(t)$ for different algorithms applied to (36), with $h = 22$ ms.

$h = [22, 40]$ ms, then $\bar{h} = [90, 370]$ ms, and $\bar{R} = [.13, .68]$. We see clearly that, in this simulation example, if prediction is affordable, the prediction-correction strategy, that is our C-FOPC algorithm is to be preferred to traditional correction-only schemes, since it achieves a lower asymptotical worst-case error. We notice that this error is lower by an half order of magnitude, while the error of the correction-extra correction and the total correction strategy are practically the same. For completeness, we report that $\boldsymbol{x}_0$ is chosen to be zero.

The result is quite remarkable, telling that performing Newton-like prediction steps on a fixed (Hessian, gradient, time derivative) triple can be computationally much more interesting that performing correction steps on a varying (i.e., re-updated) gradient.

In Figure 4, we report the time trajectories of a number of variables for the three strategies to appreciate how the constraints are in fact active.

C. An example of realistic application

As an example of realistic application, we consider the problem of optimizing the operation of aggregations of energy resources connected to a power distribution system – a research task that has gained significant interest from academic and industrial sectors because of the increased deployment of distributed energy resources (DERs). The proposed algorithm can be implemented to enable a *real-time* optimization of the DER operation, where the term "real-time" refers to a setting where the power setpoints of the DERs are updated on a second or subsecond time scale based on streaming measurements, to maximize operational objectives while coping with variability of ambient conditions and non-controllable energy assets [17], [40], [41].

Consider then a distribution network consisting of one slack bus and N nodes with controllable DERs. Let $s_n = p_n + jq_n \in \mathbb{C}$ denote the net injected complex power at node j, and let $p_0 \in \mathbb{R}$ denote the active power flow at a point of connection of the distribution system with the rest of the grid. For future developments, let $\boldsymbol{x}_n := [\Re\{s_n\}, \Im\{s_n\}]^\mathsf{T}$ collect the active and reactive setpoints of the DER at node n. To further simplify the notation, define the vector $\boldsymbol{x} := [\boldsymbol{x}_1^\mathsf{T}, \dots, \boldsymbol{x}_N^\mathsf{T}]^\mathsf{T}$. To facilitate the development of computationally-affordable algorithms, we postulate the following approximate linear relationship between $\boldsymbol{x}$ and the active powers p_0:

$$\tilde{p}_0(\boldsymbol{x}, t) = \boldsymbol{m}^\mathsf{T}(t)[\boldsymbol{x} - \boldsymbol{x}_\ell(t)] + n(t)$$
$$:= \sum_{n=1}^N \boldsymbol{m}_n^\mathsf{T}(t)[\boldsymbol{x}_n - \boldsymbol{x}_{n,\ell}(t)] + \eta(t) \qquad (39)$$

where $\boldsymbol{x}_{n,\ell}(t)$ denotes the non-controllable loads and the model parameters $\{\boldsymbol{m}_n(t) \in \mathbb{R}^2\}$ and $\eta(t) \in \mathbb{R}$ can be obtained as shown in, e.g., [17], [42]. These model parameters can be either fixed or time-varying.

Consider a discrete-time operational setting where the setpoints of the DERs are updated at time instants $t_k = hk$, with $k \in \mathbb{N}$ and $h > 0$ based on the specific implementation requirements (e.g., second, sub-second, or a few seconds), and formulate the following optimization problem per time k:

$$\min_{\boldsymbol{x} \in \mathbb{R}^{2n}} \sum_{n=1}^N C_n(\boldsymbol{x}_n; t_k) + \frac{\gamma}{2} \left(p_0^{\mathrm{set}}(t_k) - \tilde{p}_0(\boldsymbol{x}, t_k) \right)^2 \quad (40a)$$

$$\text{s. t.:} \quad \boldsymbol{x}_n \in Y_n, \ \forall \ n = 1, \dots, N \qquad (40b)$$

where $p_0^{\mathrm{set}}(t_k)$ is a setpoint for the aggregate power at the point of connection [41], $\gamma > 0$ is a design parameter that influences the ability to track the reference signal $p_0^{\mathrm{set}}(t_k)$, $C_n(\boldsymbol{x}_n; t_k) : \mathbb{R}^2 \times \mathbb{R}_+ \to \mathbb{R}$ is a local cost function, and Y_n specifies the operating region for DER n. For example, Y_n can be given by

$$Y_n = \{p_n, q_n : \underline{p}_n \leqslant p_n \leqslant \overline{p}_n, \underline{q}_n \leqslant q_n \leqslant \overline{q}_n\} \qquad (41)$$

with $\underline{p}_n$, $\overline{p}_n$, $\underline{q}_n$, and $\overline{q}_n$ given limits. This is the case, for example, for variable-speed drives, natural-gas generators, electric vehicle chargers, and renewable sources of energy operating with (fixed) headroom from the maximum available power. Problem (40) is pertinent for settings where a number of customers aim at maximizing given performance objectives

related to the DERs (e.g., minimization of net electricity payment), while partaking in grid services; specifically, customers may be incentivized by utility or aggregators to adjust the DERs' output powers so that the aggregate power $\tilde{p}_0(\boldsymbol{x}; t_k)$ can be driven to the setpoint $p_0^{\mathrm{set}}(t_k)$.

For the numerical experiments, we utilize real data belonging to residential loads from a neighborhood located in the Sacramento metro area [43]. The data have a granularity of 1 second and they are utilized to populate the IEEE 37-node test feeder (see e.g., [17]). For simplicity, we consider a single-phase equivalent of the feeder. The model parameters $\{\boldsymbol{m}_n(t) \in \mathbb{R}^2\}$ and $\eta(t) \in \mathbb{R}$ are computed as shown in [17]. With this setting, the non-controllable portion of $\tilde{p}_0(\boldsymbol{x}; t_k)$ in (40), which defined as

$$a(t_k) := -\sum_{n=1}^{N} \boldsymbol{m}_n^{\mathsf{T}}(t_k)\boldsymbol{x}_{n,\ell}(t_k) + \eta(t_k) \qquad (42)$$

varies over time as illustrated in Figure 5. The trajectory for the setpoint $p_0^{\mathrm{set}}(t_k)$ is also provided in Fig. 5; in the considered setting, the specified setpoints promote a smoother variability of the aggregate load $\tilde{p}_0(\boldsymbol{x}; t)$ over time.

With reference to the feeder in [17], we assume that controllable DERs are located at nodes 4, 7, 10, 13, 17, 20, 22, 26, 30, and 35, and it is assumed that the real power p_n can be controlled within the interval $\underline{p}_n = -50$ kW and $\overline{p}_n = 50$ kW. The cost functions $C_n(\boldsymbol{x}_n; t_k) = p_n^2$ and $\gamma = 2$. This setting is representative of the case where DERs can deviate their active powers from a nominal operating point to provide services to the grid, but their adjustment is limited within $[\underline{p}_n, \overline{p}_n]$. In addition, $\boldsymbol{m}_n(t_k)$ is considered constant in time (as often varies at slower time-scales than $a(t_k)$, typically minutes).

To track the solution trajectory $\boldsymbol{x}^*(t_k)$ of (40) while k increases, we run both a running projected gradient algorithm, i.e., Algorithm 1 with $P = 0$, and our prediction-correction Algorithm 1. The prediction step is computed by approximating the time-derivative of the gradient as explained in [29], which does not affect convergence or the rates.

For the simulation runs, the functional parameters are $m = 1$ and $L = 21$, while since the cost function is a time-varying quadratic function with constant Hessian, $C_1 = 0$, $C_2 = 0$, the convergence attraction region is the whole space and there is no upper bound on the sampling period (see Theorem 2). We choose the stepsizes α and β as $\alpha = \beta = 0.0048$, such that $\varrho_\mathrm{P} = \varrho_\mathrm{C} < 1$, while $h = 1$ s, and $\boldsymbol{x}_0 = \mathbf{0}$.

We keep the computational time fixed in our comparison of running projected gradient and prediction-correction algorithm, so that we could identify better the contribution of the prediction step. With the same notation and nomenclature of Section IV-B, we look at a total-correction strategy for the running projected gradient with $C = 3$. Each correction step takes 76 μs on a 1.8 GHz Intel Core i5 (and 3 steps, 228 μs). For the prediction-correction algorithm, we fix $C = 1$, while we choose $P = 2$, such that the total computational time is 206 μs, which is similar to the running strategy.

We report the error $\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|$ of the two strategies in time in Figure 6 (for readability purposes only for the time frame 12:38-12:48, but in fact it is qualitatively the same for all the other time frames), and in particular the averaged

Fig. 5. Aggregate effect of the non-controllable residential load on the power at the point of common coupling $\tilde{p}_0(\boldsymbol{x}; t_k)$ [kW] in the considered simulation setting. The red trajectory represents the prescribed setpoint $p_0^{\mathrm{set}}(t_k)$.

Fig. 6. Tracking error in [kW] for the two considered strategies in the time frame 12:38-12:48.

asymptotical error (averaged in the time frame 12:10-13:00) of the running method ($P = 0$) is 3.73 kW, while the one of the prediction-correction algorithm is 3.14 kW. As captured in Figure 6, the prediction-correction algorithm seems to be more reactive to changes, responding more quickly, which is expected. This further corroborates the usefulness of the prediction step, hence of the prediction-correction methodology, even when real noisy time-varying data is used, and the time derivative of the gradient can only be approximated.

V. CONCLUSIONS

We have proposed first-order algorithms to find and track the solution trajectory of strongly convex, strongly smooth constrained time-varying optimization problems. These algorithms rely on a discrete-time prediction-correction strategy, by which at each sampling time, the decision variables are corrected through one or multiple projected gradient steps, and then used to predict the next decision variables via successive projected gradient steps on a suitably defined quadratic program. The proposed algorithms exhibit better asymptotical accuracy than state-of-the-art correction-only schemes, even when computational complexity issues are taken into account.

APPENDIX A
PRELIMINARIES

We begin the convergence analysis by deriving an upper bound on the norm of the approximation error $\boldsymbol{\Delta}_k \in \mathbb{R}^n$ incurred by the Taylor expansion in (4). In particular, given the optimal solution $\boldsymbol{x}^*(t_k)$ at t_k, compute the optimal prediction step via the Taylor approximation (4) and indicate the optimal prediction as $\boldsymbol{x}^*_{k+1|k}$. The objective is to bound the error:

$$\boldsymbol{\Delta}_k := \boldsymbol{x}^*_{k+1|k} - \boldsymbol{x}^*(t_{k+1}), \qquad (43)$$

which is committed when $\boldsymbol{x}^*(t_{k+1})$ is replaced by $\boldsymbol{x}^*_{k+1|k}$ (here we use the superscript $*$ to indicate that we compute the prediction from $\boldsymbol{x}^*(t_k)$ and not any $\boldsymbol{x}_k$).

Proposition 1: **The error norm $\|\boldsymbol{\Delta}_k\|$ is upper bounded as follows.**

Case a) **Under the sole Assumption 1**

$$\|\boldsymbol{\Delta}_k\| \leqslant 2h\frac{C_0}{m}(1 + L/m) =: \Delta_1 = O(h). \qquad (44)$$

Case b) **Under both Assumption 1 and 2**

$$\|\boldsymbol{\Delta}_k\| \leqslant \frac{h^2}{2}\left[\frac{C_0^2 C_1}{m^3} + \frac{2C_0 C_2}{m^2} + \frac{C_3}{m}\right] =: \Delta_2 = O(h^2). \qquad (45)$$

Proof: Let us start by simplifying the notation. Define

$$\nabla_{\boldsymbol{x}} f_i = \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_{k+i}); t_{k+i}), \quad \mathbf{Q}_i = \nabla_{\boldsymbol{xx}} f(\boldsymbol{x}^*(t_{k+i}); t_{k+i}) \qquad (46)$$

$$\boldsymbol{c}_i = \nabla_{t\boldsymbol{x}} f(\boldsymbol{x}^*(t_{k+i}); t_{k+i}), \quad \boldsymbol{x}_i = \boldsymbol{x}^*(t_{k+i}), \ \boldsymbol{x} = \boldsymbol{x}^*_{k+1|k}. \qquad (47)$$

With this notation in place, $\boldsymbol{\Delta}_k = \boldsymbol{x} - \boldsymbol{x}_1$ [Cf. (43)]. In addition, $\boldsymbol{x}$ is computed by the generalized equation (4),

$$\nabla_{\boldsymbol{x}} f_0 + \mathbf{Q}_0(\boldsymbol{x} - \boldsymbol{x}_0) + h\,\boldsymbol{c}_0 + N_X(\boldsymbol{x}) \ni \mathbf{0}, \qquad (48)$$

while $\boldsymbol{x}_1$ is the solution of

$$\nabla_{\boldsymbol{x}} f_1 + N_X(\boldsymbol{x}_1) \ni \mathbf{0}. \qquad (49)$$

Define the supporting functions,

$$G(\boldsymbol{y}) = \nabla_{\boldsymbol{x}} f_1 + \mathbf{Q}_1(\boldsymbol{y} - \boldsymbol{x}_1) + N_X(\boldsymbol{y}) \qquad (50)$$

$$g(\boldsymbol{y}) = \nabla_{\boldsymbol{x}} f_0 + \mathbf{Q}_0(\boldsymbol{y} - \boldsymbol{x}_0) + h\,\boldsymbol{c}_0 + \\ - [\nabla_{\boldsymbol{x}} f_1 + \mathbf{Q}_1(\boldsymbol{y} - \boldsymbol{x}_1)]. \qquad (51)$$

These two functions allows one to rewrite (48) as

$$(g + G)(\boldsymbol{x}) \ni \mathbf{0}. \qquad (52)$$

It is also true that $(g + G)(\boldsymbol{x}_1) \ni g(\boldsymbol{x}_1)$, since for optimality $G(\boldsymbol{x}_1) \ni \mathbf{0}$. Define $F(\boldsymbol{y}) = (g + G)(\boldsymbol{y})$, and consider the parametric generalized equation $F(\boldsymbol{y}) + \boldsymbol{p} \ni \mathbf{0}$. Under the Assumptions 1-2, as for [34, Theorem 2F.9], we have that the solution mapping $\boldsymbol{p} \mapsto \boldsymbol{y}(\boldsymbol{p})$ for the generalized equation $F(\boldsymbol{y}) + \boldsymbol{p} \ni \mathbf{0}$ is every-where single valued and Lipschitz continuous with constant m^{-1}, that is

$$\|\boldsymbol{y}(\boldsymbol{p}) - \boldsymbol{y}(\boldsymbol{p}')\| \leqslant \frac{1}{m}\|\boldsymbol{p} - \boldsymbol{p}'\|. \qquad (53)$$

We can set $\boldsymbol{p} = \mathbf{0}$ and $\boldsymbol{p}' = -g(\boldsymbol{x}_1)$, which leads to

$$\|\boldsymbol{x} - \boldsymbol{x}_1\| = \|\boldsymbol{\Delta}_k\| \leqslant \frac{1}{m}\|g(\boldsymbol{x}_1)\|. \qquad (54)$$

We proceed now to bound $\|g(\boldsymbol{x}_1)\|$. We can write $g(\boldsymbol{x}_1)$ as

$$g(\boldsymbol{x}_1) = \nabla_{\boldsymbol{x}} f_0 + \mathbf{Q}_0(\boldsymbol{x}_1 - \boldsymbol{x}_0) + h\,\boldsymbol{c}_0 - \nabla_{\boldsymbol{x}} f_1. \qquad (55)$$

A. Case a)

Given the expression of $g(\boldsymbol{x}_1)$, we can bound its norm via Assumption 1

$$\|g(\boldsymbol{x}_1)\| \leqslant \|\nabla_{\boldsymbol{x}} f_0 - \nabla_{\boldsymbol{x}} f_1\| + \|\mathbf{Q}_0\|\|\boldsymbol{x}_1 - \boldsymbol{x}_0\| + h\,\|\boldsymbol{c}_0\| \\ \leqslant \|\nabla_{\boldsymbol{x}} f_0 - \nabla_{\boldsymbol{x}} f_1\| + L\|\boldsymbol{x}_1 - \boldsymbol{x}_0\| + hC_0, \qquad (56)$$

where we have used the upper bound on the Hessian of the cost function. Furthermore,

$$\|\nabla_{\boldsymbol{x}} f_0 - \nabla_{\boldsymbol{x}} f_1\| = \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_k) - \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_{k+1}); t_{k+1})\| \\ \leqslant \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_{k+1}) - \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_{k+1}); t_{k+1})\| + \\ \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_k) - \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_{k+1})\| \qquad (57)$$

We can bound the first term in the right hand side with the Lipschitz property of the gradient of the cost function as

$$\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_{k+1}) - \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_{k+1}); t_{k+1})\| \leqslant L\|\boldsymbol{x}_1 - \boldsymbol{x}_0\|, \qquad (58)$$

while the second part can be bounded by the upper bound on $\nabla_{t\boldsymbol{x}} f$, as

$$\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_k) - \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_{k+1})\| = \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_k) \\ - \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_k) - h\,\nabla_{t\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); \tau)\| \leqslant hC_0, \qquad (59)$$

where we have used the mean value theorem and $\tau \in [t_k, t_{k+1}]$. By putting Eq.s (56), (57), (58), and (59) together, one arrives at

$$\|g(\boldsymbol{x}_1)\| \leqslant 2L\|\boldsymbol{x}_1 - \boldsymbol{x}_0\| + 2h\,C_0, \qquad (60)$$

and by using (54) as well as the bound on $\|\boldsymbol{x}_1 - \boldsymbol{x}_0\|$ in (14), the claim (44) follows.

B. Case b)

By allowing higher order smoothness, the bound on $g(\boldsymbol{x}_1)$ can be improved as follows. Consider (55): it is nothing else but the error of the truncated Taylor expansion of $\nabla_{\boldsymbol{x}} f_1$:

$$\nabla_{\boldsymbol{x}} f_1 - \nabla_{\boldsymbol{x}} f_0 = \mathbf{Q}_0(\boldsymbol{x}_1 - \boldsymbol{x}_0) + h\boldsymbol{c}_0 + \boldsymbol{\epsilon}, \qquad (61)$$

where the error $\boldsymbol{\epsilon}$ can be bounded as

$$\|\boldsymbol{\epsilon}\| \leqslant \frac{1}{2}\Big(\|\nabla_{\boldsymbol{xxx}} f\|\|\boldsymbol{x}_1 - \boldsymbol{x}_0\|^2 + h\,\|\nabla_{t\boldsymbol{xx}} f\|\|\boldsymbol{x}_1 - \boldsymbol{x}_0\| + \\ h\,\|\nabla_{\boldsymbol{x}t\boldsymbol{x}} f\|\|\boldsymbol{x}_1 - \boldsymbol{x}_0\| + h^2\|\nabla_{tt\boldsymbol{x}} f\|\Big), \qquad (62)$$

and by using the upper bounds in Assumption 2,

$$\|g(\boldsymbol{x}_1)\| \leqslant \frac{1}{2}\,C_1\|\boldsymbol{x}_1 - \boldsymbol{x}_0\|^2 + h\,C_2\|\boldsymbol{x}_1 - \boldsymbol{x}_0\| + \frac{1}{2}h^2\,C_3. \qquad (63)$$

By using the bound (14) on the variability of the optimizers $\boldsymbol{x}_1$ and $\boldsymbol{x}_0$, then

$$\|g(\boldsymbol{x}_1)\| \leqslant h^2\,C_1\frac{C_0^2}{2m^2} + h^2\,C_2\frac{C_0}{m} + \frac{1}{2}h^2\,C_3, \qquad (64)$$

and by combining this bound with (54), the claim (45) follows. ∎

APPENDIX B
PROOF OF THEOREMS 1 AND 2

We divide the proof in different steps. Step 1: we bound the prediction error in Propositions 2 and 3; Step 2: we bound the correction error; Step 3: we put the previous steps together and derive the convergence requirements and results. As done for Proposition 1, Case a) will refer to Theorem 1, while Case b) to Theorem 2.

Prediction error. The optimal prediction error, i.e., the distance between the optimal predicted variable $\boldsymbol{x}_{k+1|k}$ and the optimizer at time step t_{k+1}, $\boldsymbol{x}^*(t_{k+1})$ can be bounded as the following proposition.

Proposition 2: **The following facts hold true.**

Case a) Under the same assumptions and notation of Theorem 1, we have that

$$\|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| \leqslant \frac{2L}{m}\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \Delta_1. \quad (65)$$

Case b) Under the same assumptions and notation of Theorem 2, we have that

$$\|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| \leqslant \frac{C_1}{2\,m}\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|^2 +$$
$$h\left(\frac{C_1 C_0}{m^2} + \frac{C_2}{m}\right)\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \Delta_2. \quad (66)$$

Proof: We proceed as in the proof of Proposition 1. We use similar simplifications of (46), as

$$\nabla_{\boldsymbol{x}}f_k = \nabla_{\boldsymbol{x}}f(\boldsymbol{x}_k; t_k), \quad \mathbf{Q}_k = \nabla_{\boldsymbol{xx}}f\boldsymbol{x}_k; t_k) \quad (67\text{a})$$
$$\boldsymbol{c}_k = \nabla_{t\boldsymbol{x}}f(\boldsymbol{x}_k; t_k), \quad \boldsymbol{x} = \boldsymbol{x}_{k+1|k}. \quad (67\text{b})$$

while $\nabla_{\boldsymbol{x}}f_1$ and $\boldsymbol{x}_1$ are defined just as in (46). The error $\|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\|$ is now $\boldsymbol{x} - \boldsymbol{x}_1$.

The vector $\boldsymbol{x}$ is computed by the generalized equation (4),

$$\nabla_{\boldsymbol{x}}f_k + \mathbf{Q}_k(\boldsymbol{x} - \boldsymbol{x}_k) + h\,\boldsymbol{c}_k + N_X(\boldsymbol{x}) \ni \mathbf{0}, \quad (68)$$

while $\boldsymbol{x}_1$ is the solution of (49).

Define the supporting functions,

$$G(\boldsymbol{y}) = \nabla_{\boldsymbol{x}}f_1 + \mathbf{Q}_1(\boldsymbol{y} - \boldsymbol{x}_1) + N_X(\boldsymbol{y}) \quad (69)$$
$$g(\boldsymbol{y}) = \nabla_{\boldsymbol{x}}f_k + \mathbf{Q}_k(\boldsymbol{y} - \boldsymbol{x}_k) + h\,\boldsymbol{c}_k +$$
$$- [\nabla_{\boldsymbol{x}}f_1 + \mathbf{Q}_1(\boldsymbol{y} - \boldsymbol{x}_1)]. \quad (70)$$

These two functions allows one to rewrite (68) as

$$(g + G)(\boldsymbol{x}) \ni \mathbf{0}. \quad (71)$$

It is also true that $(g + G)(\boldsymbol{x}_1) \ni g(\boldsymbol{x}_1)$, since for optimality $G(\boldsymbol{x}_1) \ni \mathbf{0}$. Define $F(\boldsymbol{y}) = (g + G)(\boldsymbol{y})$, and consider the parametric generalized equation $F(\boldsymbol{y}) + \boldsymbol{p} \ni \mathbf{0}$. Due to Assumptions 1-2, and due to [34, Theorem 2F.9], we have that the solution mapping $\boldsymbol{p} \mapsto \boldsymbol{y}(\boldsymbol{p})$ of the generalized equation $F(\boldsymbol{y}) + \boldsymbol{p} \ni \mathbf{0}$ is every-where single valued and Lipschitz continuous as

$$\|\boldsymbol{y}(\boldsymbol{p}) - \boldsymbol{y}(\boldsymbol{p}')\| \leqslant \frac{1}{m}\|\boldsymbol{p} - \boldsymbol{p}'\|. \quad (72)$$

Set $\boldsymbol{p} = \mathbf{0}$ and $\boldsymbol{p}' = -g(\boldsymbol{x}_1)$, then,

$$\|\boldsymbol{x} - \boldsymbol{x}_1\| \leqslant \frac{1}{m}\|g(\boldsymbol{x}_1)\|. \quad (73)$$

We proceed now to bound $\|g(\boldsymbol{x}_1)\|$. We can write $g(\boldsymbol{x}_1)$ as

$$g(\boldsymbol{x}_1) = \nabla_{\boldsymbol{x}}f_k + \mathbf{Q}_k(\boldsymbol{x}_1 - \boldsymbol{x}_k) + h\,\boldsymbol{c}_k - \nabla_{\boldsymbol{x}}f_1, \quad (74)$$

As in Proposition 1, we have two cases:

$$\text{Case a)} \quad \|g(\boldsymbol{x}_1)\| \leqslant 2L\|\boldsymbol{x}_1 - \boldsymbol{x}_k\| + 2h\,C_0, \quad (75)$$
$$\text{Case b)} \quad \|g(\boldsymbol{x}_1)\| \leqslant \frac{1}{2}\,C_1\|\boldsymbol{x}_1 - \boldsymbol{x}_k\|^2 +$$
$$h\,C_2\|\boldsymbol{x}_1 - \boldsymbol{x}_k\| + \frac{1}{2}h^2\,C_3. \quad (76)$$

Since $\|\boldsymbol{x}_1 - \boldsymbol{x}_k\| \leqslant \|\boldsymbol{x}_1 - \boldsymbol{x}^*(t_k)\| + \|\boldsymbol{x}^*(t_k) - \boldsymbol{x}_k\|$, and we can bound the first term of the right-hand side by using (14), then

$$\text{Case a)} \, \|g(\boldsymbol{x}_1)\| \leqslant 2L\|\boldsymbol{x}^*(t_k) - \boldsymbol{x}_k\| + 2hC_0(1 + L/m), \quad (77)$$

$$\text{Case b)} \, \|g(\boldsymbol{x}_1)\| \leqslant h^2\,C_1\frac{C_0^2}{2m^2} + h^2\,C_2\frac{C_0}{2m} + \frac{1}{2}h^2\,C_3 +$$
$$\frac{1}{2}C_1\|\boldsymbol{x}^*(t_k) - \boldsymbol{x}_k\|^2 + h\left(\frac{C_1 C_0}{m} + C_2\right)\|\boldsymbol{x}^*(t_k) - \boldsymbol{x}_k\|. \quad (78)$$

By combining these two last bounds with (73), the claims (65)-(66) follow. ∎

On the other hand, the distance between the approximate prediction $\tilde{\boldsymbol{x}}_{k+1|k}$ and the optimal prediction $\boldsymbol{x}_{k+1|k}$ can be bounded by using standard results for the projected gradient method, for the proof see for instance [44], [45].

Proposition 3: **Under Assumption 1, we have that**

$$\|\tilde{\boldsymbol{x}}_{k+1|k} - \boldsymbol{x}_{k+1|k}\| \leqslant \varrho_{\mathrm{P}}^P\|\boldsymbol{x}_k - \boldsymbol{x}_{k+1|k}\|, \quad (79)$$

with $\varrho_{\mathrm{P}} = \max\{|1 - \alpha m|, |1 - \alpha L|\}$**.**

By putting together Propositions 2 and 3 and (14) , we obtain for the total error after prediction as

$$\|\tilde{\boldsymbol{x}}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| \leqslant \|\tilde{\boldsymbol{x}}_{k+1|k} - \boldsymbol{x}_{k+1|k}\| + \|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| \quad (80\text{a})$$
$$\leqslant \varrho_{\mathrm{P}}^P\|\boldsymbol{x}_k - \boldsymbol{x}_{k+1|k}\| + \|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| \quad (80\text{b})$$
$$\leqslant \varrho_{\mathrm{P}}^P\left(\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \|\boldsymbol{x}^*(t_k) - \boldsymbol{x}^*(t_{k+1})\| +$$
$$\|\boldsymbol{x}^*(t_{k+1}) - \boldsymbol{x}_{k+1|k}\|\right) + \|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| \quad (80\text{c})$$
$$\leqslant \varrho_{\mathrm{P}}^P\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \left(\varrho_{\mathrm{P}}^P + 1\right)\|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| +$$
$$\varrho_{\mathrm{P}}^P\|\boldsymbol{x}^*(t_{k+1}) - \boldsymbol{x}^*(t_k)\| \quad (80\text{d})$$
$$\leqslant \eta_0\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|^2 + \eta_1\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \eta_2, \quad (80\text{e})$$

where we have defined

$$\text{Case a)} \quad \begin{cases} \eta_0 &= 0, \\ \eta_1 &= \varrho_{\mathrm{P}}^P + (\varrho_{\mathrm{P}}^P + 1)\frac{2L}{m}, \\ \eta_2 &= (2\varrho_{\mathrm{P}}^P + 1)\Delta_1, \end{cases} \quad (81)$$

$$\text{Case b)} \quad \begin{cases} \eta_0 &= (\varrho_{\mathrm{P}}^P + 1)\frac{C_1}{2\,m}, \\ \eta_1 &= \varrho_{\mathrm{P}}^P + h\,(\varrho_{\mathrm{P}}^P + 1)\left(\frac{C_1 C_0}{m^2} + \frac{C_2}{m}\right), \\ \eta_2 &= \varrho_{\mathrm{P}}^P\left(h\,\frac{C_0}{m} + \Delta_2\right) + \Delta_2, \end{cases} \quad (82)$$

where Δ_1 and Δ_2 are defined as in Proposition 1.

Correction error. We look now at the correction step, which by using standard results for the projected gradient method, we have

$$\|\boldsymbol{x}_{k+1} - \boldsymbol{x}^*(t_{k+1})\| \leqslant \varrho_{\mathrm{C}}^C\|\tilde{\boldsymbol{x}}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\|, \quad (83)$$

with $\varrho_C = \max\{|1-\alpha m|, |1-\alpha L|\}$. And by putting together the result (80e) with (83), we obtain the recursive error bound,

$$\|\boldsymbol{x}_{k+1} - \boldsymbol{x}^*(t_{k+1})\| \leqslant \varrho_C^C \Big(\eta_0 \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|^2 + $$
$$\eta_1 \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \eta_2 \Big). \quad (84)$$

Global error and convergence.

We start with Case a). Call for simplicity $\bar{\eta}_i = \varrho_C^C \eta_i$, for $i = 0, 1, 2$. Since $\eta_0 = 0$, then convergence is achieved if

$$\bar{\eta}_1 < 1 \iff \tau_0 = \varrho_C^C \left[\varrho_P^P + (\varrho_P^P + 1) \frac{2L}{m} \right] < 1, \quad (85)$$

which guarantees a monotonical decrease of the optimality gap with linear rate and asymptotic bound of

$$\limsup_{k \to \infty} \|\boldsymbol{x}_{k+1} - \boldsymbol{x}^*(t_{k+1})\| = \frac{\varrho_C^C \Delta_1}{1 - \tau_0} (2\varrho_P^P + 1) = O(\varrho_C^C h), \quad (86)$$

which are the claims in Theorem 1.

We finish with Case b). Call again for simplicity $\bar{\eta}_i = \varrho_C^C \eta_i$, for $i = 0, 1, 2$. Then convergence is achieved if

1) Each iteration does not increase the error, so that

$$\bar{\eta}_0 \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|^2 + \bar{\eta}_1 \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \bar{\eta}_2 \leqslant$$
$$\tau \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \bar{\eta}_2 \quad (87)$$

for a $\tau < 1$;

2) One can find a $\tau < 1$ such that (87) holds.

By simple algebra, convergence is achieved if $\bar{\eta}_1 < \tau < 1$, that is if

$$\varrho_P^P \varrho_C^C < \tau, \quad h < \frac{\tau - \varrho_C^C \varrho_P^P}{\varrho_C^C (\varrho_P^P + 1)} \left(\frac{C_1 C_0}{m^2} + \frac{C_2}{m} \right)^{-1} = \bar{h}, \quad (88)$$

which sets the bounds on the number of prediction and correction steps as well as the sampling period, and if the initial optimality gap is chosen as

$$\|\boldsymbol{x}_0 - \boldsymbol{x}^*(t_0)\| \leqslant \frac{\tau - \bar{\eta}_1}{\bar{\eta}_0} = \bar{R}. \quad (89)$$

The convergence region depends on the sampling period and on the prediction and correction steps. When $h \to 0$, then

$$\lim_{h \to 0} \bar{R} = \frac{2m}{C_1} \frac{\tau - \varrho_P^P \varrho_C^C}{\varrho_C^C (\varrho_P^P + 1)}. \quad (90)$$

As for the convergence asymptotical error, by using (84) in combination with (87), we can show that

$$\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| \leqslant \tau^k \|\boldsymbol{x}_0 - \boldsymbol{x}^*(t_0)\| + \bar{\eta}_2 \frac{1 - \tau^k}{1 - \tau}, \quad (91)$$

from which, by letting $k \to \infty$, the result (21) follows. ∎

APPENDIX C
PROOF OF THEOREM 4

We proceed similarly to Appendix B. One of the main difference is a new Proposition 2 suited for the situation at hand.

Proposition 4: **The following facts hold true.**

Case a) Under the same assumptions and notation of Theorem 3, we have that

$$\|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| \leqslant \left(1 - \gamma + \gamma \frac{2L}{m} \right) \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + 2\Delta_1. \quad (92)$$

Case b) Under the same assumptions and notation of Theorem 4, we have that

$$\|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| \leqslant \gamma \frac{C_1}{2m} \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|^2 +$$
$$\left[1 - \gamma + h \left(\frac{C_1 C_0}{m^2} + \frac{C_2}{m} \right) \right] \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \Delta_2. \quad (93)$$

Proof: Start by noticing that Proposition 1 holds true even when $\gamma < 1$ (in the unconstrained case), since $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t_k); t_k) = \nabla_{\boldsymbol{x}} f_0 = \boldsymbol{0}$. In fact, Eq. (48) should read

$$\nabla_{\boldsymbol{x}} f_0 + \mathbf{Q}_0 (\boldsymbol{x} - \boldsymbol{x}_0) + h\, \boldsymbol{c}_0 = (1 - \gamma) \nabla_{\boldsymbol{x}} f_0, \quad (94)$$

but this is in fact equivalent to the original (48), since $\nabla_{\boldsymbol{x}} f_0 = \boldsymbol{0}$, and therefore the whole proposition is still valid, and in particular $\|\boldsymbol{x}_{k+1|k}^* - \boldsymbol{x}^*(t_{k+1})\| \leqslant \Delta_i$, where $i = 1$ for Case a) and $i = 2$ for Case b).

Call now $\delta \boldsymbol{x}_k = \boldsymbol{x}_{k+1|k} - \boldsymbol{x}_k$, and $\delta \boldsymbol{x}_k^* = \boldsymbol{x}_{k+1|k}^* - \boldsymbol{x}^*(t_k)$. Then,

$$\|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| =$$
$$\|\boldsymbol{x}_k + \delta \boldsymbol{x}_k - (\boldsymbol{x}^*(t_k) + \delta \boldsymbol{x}_k^*) + (\boldsymbol{x}_{k+1|k}^* - \boldsymbol{x}^*(t_{k+1}))\|, \quad (95)$$

which given Proposition 1 and by using the Triangle inequality can be upper bounded as

$$\|\boldsymbol{x}_{k+1|k} - \boldsymbol{x}^*(t_{k+1})\| \leqslant (1 - \gamma) \|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \|\delta \check{\boldsymbol{x}}_k - \delta \boldsymbol{x}_k^*\| + \Delta_i, \quad (96)$$

where we have set $\delta \check{\boldsymbol{x}}_k = \delta \boldsymbol{x}_k + \gamma(\boldsymbol{x}_k - \boldsymbol{x}^*(t_k))$. Note that this decomposition may seem cumbersome, yet it is the cornerstone of the proof of this proposition.

For $\delta \check{\boldsymbol{x}}_k$ and $\delta \boldsymbol{x}_k^*$, it holds that

$$\gamma \nabla_{\boldsymbol{x}} f_k + \gamma \mathbf{Q}_k (\boldsymbol{x}_0 - \boldsymbol{x}_k) + \mathbf{Q}_k \delta \check{\boldsymbol{x}}_k + h\, \boldsymbol{c}_k = \boldsymbol{0}, \quad (97)$$
$$\gamma \nabla_{\boldsymbol{x}} f_0 + \mathbf{Q}_0 \delta \boldsymbol{x}_k^* + h\, \boldsymbol{c}_0 = \boldsymbol{0}, \quad (98)$$

where as in (67) we have used the simplifications

$$\nabla_{\boldsymbol{x}} f_k = \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k; t_k), \quad \mathbf{Q}_k = \nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k) \quad (99a)$$
$$\boldsymbol{c}_k = \nabla_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k), \quad \boldsymbol{x} = \boldsymbol{x}_{k+1|k}. \quad (99b)$$

while $\nabla_{\boldsymbol{x}} f_0$, $\mathbf{Q}_0$, $\boldsymbol{c}_0$, and $\boldsymbol{x}_0$ are defined just as in (46)-(47), for $i = 0$ (i.e., $\boldsymbol{x}_0 = \boldsymbol{x}^*(t_k)$, and so on). Define

$$g(\delta \boldsymbol{x}) = \gamma (\nabla_{\boldsymbol{x}} f_k - \nabla_{\boldsymbol{x}} f_0 + \mathbf{Q}_k (\boldsymbol{x}_0 - \boldsymbol{x}_k)) +$$
$$(\mathbf{Q}_k - \mathbf{Q}_0) \delta \boldsymbol{x} + h (\boldsymbol{c}_k - \boldsymbol{c}_0), \quad (100)$$
$$G(\delta \boldsymbol{x}) = \gamma \nabla_{\boldsymbol{x}} f_0 + \mathbf{Q}_0 \delta \boldsymbol{x} + h\, \boldsymbol{c}_0, \quad (101)$$

and notice that $(g + G)(\delta \check{\boldsymbol{x}}_k) = \boldsymbol{0}$, while $(g + G)(\delta \boldsymbol{x}_k^*) = g(\delta \boldsymbol{x}_k^*)$. With a similar argument as the one of the proof of Proposition 2, then

$$\|\delta \check{\boldsymbol{x}}_k - \delta \boldsymbol{x}_k^*\| \leqslant \frac{1}{m} \|g(\delta \boldsymbol{x}_k^*)\|. \quad (102)$$

Let us now bound $\|g(\delta \boldsymbol{x}_k^*)\|$.

Case a) Due to Assumptions 1:

$$\|g(\delta \boldsymbol{x}_k^*)\| \leqslant 2\gamma L \|\boldsymbol{x}_k - \boldsymbol{x}_0\| + 2L \|\delta \boldsymbol{x}_k^*\| + 2h\, C_0. \quad (103)$$

Case b) Due to Assumptions 1-2,

$$\|g(\delta \boldsymbol{x}_k^*)\| \leqslant \gamma \frac{C_1}{2}\|\boldsymbol{x}_k - \boldsymbol{x}_0\|^2 + C_1\|\boldsymbol{x}_k - \boldsymbol{x}_0\|\|\delta \boldsymbol{x}_k^*\| + h\,C_2\|\boldsymbol{x}_k - \boldsymbol{x}_0\|. \tag{104}$$

The next step of the proof is to upper bound the term $\|\delta \boldsymbol{x}_k^*\| = \|\boldsymbol{x}_{k+1|k}^* - \boldsymbol{x}^*(t_k)\|$. We know that

$$\gamma \nabla_{\boldsymbol{x}} f_0 + \mathbf{Q}_0 \delta \boldsymbol{x}_k^* + h\boldsymbol{c}_0 = \mathbf{0}, \tag{105}$$

and since $\nabla_{\boldsymbol{x}} f_0 = \mathbf{0}$, then $\delta \boldsymbol{x}_k^* = -h\,\mathbf{Q}_0^{-1}\boldsymbol{c}_0$. Which yields

$$\|\delta \boldsymbol{x}_k^*\| \leqslant h\,\frac{C_0}{m}. \tag{106}$$

By putting together the bounds (96), (102), (103), (104), and (106), the claims (92) and (93) follow. ∎

By using Proposition 4 along with the same arguments as the one in Eq.s (79) till (84), we arrive at the error recursion

$$\|\boldsymbol{x}_{k+1} - \boldsymbol{x}^*(t_{k+1})\| \leqslant \bar{\eta}_0'\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\|^2 + \bar{\eta}_1'\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| + \bar{\eta}_2', \tag{107}$$

where,

$$\text{Case a)} \begin{cases} \bar{\eta}_0' &= 0, \\ \bar{\eta}_1' &= \varrho_{\mathsf{C}}^C\left[\varrho_{\mathsf{P}}^P + (\varrho_{\mathsf{P}}^P + 1)\left(1 - \gamma + \gamma \frac{2L}{m}\right)\right], \\ \bar{\eta}_2' &= \varrho_{\mathsf{C}}^C\left[2(2\varrho_{\mathsf{P}}^P + 1)\Delta_1\right], \end{cases} \tag{108}$$

Case b)

$$\begin{cases} \bar{\eta}_0' &= \gamma\varrho_{\mathsf{C}}^C(\varrho_{\mathsf{P}}^P + 1)\frac{C_1}{2\,m}, \\ \bar{\eta}_1' &= \varrho_{\mathsf{C}}^C\left[\varrho_{\mathsf{P}}^P + (\varrho_{\mathsf{P}}^P + 1)\left(1 - \gamma + h\left(\frac{C_1 C_0}{m^2} + \frac{C_2}{m}\right)\right)\right], \\ \bar{\eta}_2' &= \varrho_{\mathsf{C}}^C\left[\varrho_{\mathsf{P}}^P\left(h\,\frac{C_0}{m} + \Delta_2\right) + \Delta_2\right]. \end{cases} \tag{109}$$

By using now the same reasoning as in Eq.s (87) till Eq.s (91), the claims (26)-(30) can be proven. ∎

References

[1] A. Simonetto and E. Dall'Anese, "A First -Order Prediction-Correction Algorithm for Time-Varying (Constrained) Optimization," in *Proceedings of the 20th IFAC World Congress (to appear)*, (Toulouse, France), July 2017.

[2] J. L. Jerez, P. J. Goulart, S. Richter, G. A. Constantinides, E. C. Kerrigan, and M. Morari, "Embedded Online Optimization for Model Predictive Control at Megahertz Rates," *IEEE Transactions on Automatic Control*, vol. 59, no. 12, pp. 3238 – 3251, 2014.

[3] J.-H. Hours and C. N. Jones, "A Parametric Non-Convex Decomposition Algorithm for Real-Time and Distributed NMPC," *IEEE Transactions on Automatic Control*, vol. 61, no. 2, pp. 287 – 302, 2016.

[4] B. Gutjahr, L. Gröll, and M. Werling, "Lateral Vehicle Trajectory Optimization Using Constrained Linear Time-Varying MPC," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, no. 99, pp. 1 – 10, 2016.

[5] F. Y. Jakubiec and A. Ribeiro, "D-MAP: Distributed Maximum a Posteriori Probability Estimation of Dynamic Systems," *IEEE Transactions on Signal Processing*, vol. 61, no. 2, pp. 450 – 466, 2013.

[6] M. S. Asif and J. Romberg, "Sparse recovery of streaming signals using ℓ_1-homotopy ," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4209 – 4223, 2014.

[7] Y. Yang, M. Zhang, M. Pesavento, and D. P. Palomar, "An Online Parallel and Distributed Algorithm for Recursive Estimation of Sparse Signals," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 3, pp. 290 – 305, 2016.

[8] N. Vaswani and J. Zhan, "Recursive Recovery of Sparse Signal Sequences from Compressive Measurements: A Review," *IEEE Transactions on Signal Processing*, vol. 64, no. 13, pp. 3523 – 3549, 2016.

[9] A. Balavoine, J. Romberg, and C. Rozell, "Discrete and continuous iterative soft thresholding with a dynamic input," *IEEE Transactions on Signal Processing*, vol. 63, no. 12, pp. 3165 – 3176, 2015.

[10] A. Simonetto and G. Leus, "On Non-Differentiable Time-Varying Optimization," in *Proceedings of the 6th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, (Cancun, Mexico), December 2015.

[11] P. Sopasakis, N. Freris, and P. Patrinos, "Accelerated Reconstruction of a Compressively Sampled Data Stream," in *Proceedings of the 24th European Signal Processing Conference*, (Budapest, Hungary), pp. 1078 – 1082, September 2016.

[12] B. Baingana, P. Traganitis, G. Mateos, and G. B. Giannaki, *Big Data Analytics for Social Networks,* in Graph Analysis for Social Media. CRC Press, 2015.

[13] T. Ardeshiri, M. Norrlöf, J. Löfberg, and A. Hansson, "Convex Optimization Approach for Time-Optimal Path Tracking of Robots with Speed Dependent Constraints," in *Proceedings of the 18th IFAC World Congress*, (Milano, Italy), pp. 14648 – 14653, September 2011.

[14] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl, "Time-Optimal Path Tracking for Robots: a Convex Optimization Approach," *IEEE Transactions on Automatic Control*, vol. 54, no. 10, pp. 2318 – 2327, 2009.

[15] A. Koppel, G. Warnell, E. Stumpe, and A. Ribeiro, "D4L: Decentralized Dynamic Discriminative Dictionary Learning," *IEEE Transactions Signal and Information Processing over Networks*, vol. (submitted), June 2016.

[16] C. Zhao, U. Topcu, N. Li, and S. Low, "Design and Stability of Load-Side Primary Frequency Control in Power Systems," *IEEE Transactions on Automatic Control*, 2014. to appear.

[17] E. Dall'Anese and A. Simonetto, "Optimal Power Flow Pursuit," *IEEE Transactions on Smart Grid, in press*, 2016.

[18] E. Dall'Anese, A. Simonetto, and S. V. Dhople, "Design of Distributed Controllers Seeking Optimal Power Flow Solutions Under Communications Constraints," in *Proceedings of the 55th IEEE Conference on Decision and Controls*, (Las Vegas, NV, US), December 2016.

[19] A. L. Dontchev, M. I. Krastanov, R. T. Rockafellar, and V. M. Veliov, "An Euler-Newton Continuation method for Tracking Solution Trajectories of Parametric Variational Inequalities," *SIAM Journal of Control and Optimization*, vol. 51, no. 51, pp. 1823 – 1840, 2013.

[20] M. Uecker, S. Zhang, D. Voit, K.-D. Merboldt, and J. Frahm, "Real-time MRI: recent advances using radial FLASH," *Imaging Medicine*, vol. 4, no. 4, pp. 461 – 476, 2012.

[21] A. Nagurney and J. Pan, "Evolution Variational Inequalities and Projected Dynamical Systems with Application to Human Migration," *Mathematical and Computer Modelling*, vol. 43, no. 5 – 6, pp. 646 – 657, 2006.

[22] M. Ye and G. Hu, "Distributed Optimization for Systems with Time-Varying Quadratic Objective Functions," in *Proceedings of the 54th IEEE Conference on Decision and Control*, (Osaka, Japan), pp. 3285 – 3290, December 2015.

[23] S. Rahili, W. Ren, and P. Lin, "Distributed Convex Optimization of Time-Varying Cost Functions for Double-integrator Systems Using Nonsmooth Algorithms," in *Proceedings of the American Control Conference*, (Chicago (IL), USA), pp. 68 – 73, July 2015.

[24] S. Rahili and W. Ren, "Distributed Convex Optimization for Continuous-Time Dynamics with Time-Varying Cost Functions," *IEEE Transactions on Automatic Control (to appear)*, 2016.

[25] P. Gong, F. Chen, and W. Lan, "Time-varying convex optimization for double-integrator dynamics over a directed network," in *Proceedings of the 35th Chinese Control Conference*, (Chengdu, China), pp. 7341 – 7346, July 2016.

[26] M. Fazlyab, S. Paternain, V. Preciado, and A. Ribeiro, "Interior Point Method for Dynamic Constrained Optimization in Continuous Time," in *Proceedings of the American Control Conference*, (Boston (MA), USA), pp. 5612 – 5618, July 2016.

[27] M. Fazlyab, S. Paternain, V. Preciado, and A. Ribeiro, "Prediction-Correction Interior-Point Method for Time-Varying Convex Optimization," *arXiv: 1608.07544*, 2016.

[28] M. Fazlyab, C. Nowzari, G. J. Pappas, A. Ribeiro, and V. M. Preciado, "Self-Triggered Time-Varying Convex Optimization," in *Proceedings of the 55th IEEE Conference on Decision and Control*, (Las Vegas, NV, US), pp. 3090 – 3097, December 2016.

[29] A. Simonetto, A. Mokhtari, A. Koppel, G. Leus, and A. Ribeiro, "A Class of Prediction-Correction Methods for Time-Varying Convex Optimization," *IEEE Transactions on Signal Processing*, vol. 64, no. 17, pp. 4576 – 4591, 2016.

[30] A. Simonetto, A. Koppel, A. Mokhtari, G. Leus, and A. Ribeiro, "Decentralized Prediction-Correction Methods for Networked Time-Varying Convex Optimization," *IEEE Transactions on Automatic Control (to appear)*, 2017.

[31] B. T. Polyak, *Introduction to Optimization*. Optimization Software, Inc., 1987.

[32] A. Y. Popkov, "Gradient Methods for Nonstationary Unconstrained Optimization Problems," *Automation and Remote Control*, vol. 66, no. 6, pp. 883 – 891, 2005. Translated from Avtomatika i Telemekhanika, No. 6, 2005, pp. 38 – 46.

[33] S. M. Robinson, "Strongly Regular Generalized Equations," *Mathematics of Operations Research*, vol. 5, no. 1, pp. 43 – 62, 1980.

[34] A. L. Dontchev and R. T. Rockafellar, *Implicit Functions and Solution Mappings*. Springer, 2009.

[35] V. M. Zavala and M. Anitescu, "Real-Time Nonlinear Optimization as a Generalized Equation," *SIAM Journal of Control and Optimization*, vol. 48, no. 8, pp. 5444 – 5467, 2010.

[36] E. L. Allgower and K. Georg, *Numerical Continuation Methods: An Introduction*. Springer-Verlag, 1990.

[37] M. G. Cojocaru, P. Daniele, and A. Nagurney, "Projected Dynamical Systems and Evolutionary Variational Inequalities via Hilbert Spaces with Applications," *Journal of Optimization Theory and Applications*, vol. 127, no. 3, pp. 549 – 563, 2005.

[38] Y. Nesterov, "Towards Non-symmetric Conic Optimization," *Optimization Methods and Software*, vol. 27, no. 4 – 5, pp. 893 – 917, 2012.

[39] Q. Ling and A. Ribeiro, "Decentralized Dynamic Optimization Through the Alternating Direction Method of Multipliers," *IEEE Transactions on Signal Processing*, vol. 62, no. 5, pp. 1185 – 1197, 2014.

[40] L. Gan and S. H. Low, "An online gradient algorithm for optimal power flow in radial networks," *IEEE J. on Sel. Areas in Commun.*, vol. 34, no. 3, pp. 625–638, 2016.

[41] A. Bernstein, N. J. Bouman, and J.-Y. Le Boudec, "Design of resource agents with guaranteed tracking properties for real-time control of electrical grids," [Online] Available at: https://arxiv.org/abs/1511.08628.

[42] S. Bolognani and F. Dörfler, "Fast power system analysis via implicit linearization of the power flow manifold," in *2015 53rd Annual Allerton Conf. on Communication, Control, and Computing*, pp. 402–409, 2015.

[43] J. Bank and J. Hambrick, "Development of a high resolution, real time, distribution-level metering system and associated visualization modeling, and data analysis functions," National Renewable Energy Laboratory, Tech. Rep. NREL/TP-5500-56610, May 2013.

[44] E. K. Ryu and S. Boyd, "Primer on Monotone Operator Methods," *Applied Computational Mathematics*, no. 15, pp. 3 – 43, 2016.

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