



A First-order Prediction-Correction Algorithm for Time-varying (Constrained) Optimization

Preprint

Andrea Simonetto
Université catholique de Louvain

Emiliano Dall'Anese
National Renewable Energy Laboratory

*20th World Congress of the International Federation of Automatic Control
Toulouse, France
July 9–14, 2017*

**NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC**

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Conference Paper
NREL/CP-5D00-68136
July 2017

Contract No. DE-AC36-08GO28308

NOTICE

The submitted manuscript has been offered by an employee of the Alliance for Sustainable Energy, LLC (Alliance), a contractor of the US Government under Contract No. DE-AC36-08GO28308. Accordingly, the US Government and Alliance retain a nonexclusive royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for US Government purposes.

This report was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or any agency thereof.

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Available electronically at SciTech Connect <http://www.osti.gov/scitech>

Available for a processing fee to U.S. Department of Energy and its contractors, in paper, from:

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831-0062
OSTI <http://www.osti.gov>
Phone: 865.576.8401
Fax: 865.576.5728
Email: reports@osti.gov

Available for sale to the public, in paper, from:

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312
NTIS <http://www.ntis.gov>
Phone: 800.553.6847 or 703.605.6000
Fax: 703.605.6900
Email: orders@ntis.gov

Cover Photos by Dennis Schroeder: (left to right) NREL 26173, NREL 18302, NREL 19758, NREL 29642, NREL 19795.

NREL prints on paper that contains recycled content.

A First-order Prediction-Correction Algorithm for Time-varying (Constrained) Optimization

Andrea Simonetto* Emiliano Dall’Anese**

* *Université catholique de Louvain, Louvain-la-Neuve, Belgium*
(e-mail: andrea.simonetto@uclouvain.be)

** *National Renewable Energy Laboratory, Golden, CO, USA*
(e-mail: emiliano.dallanese@nrel.gov)

Abstract: This paper focuses on the design of online algorithms based on prediction-correction steps to track the optimal solution of a time-varying constrained problem. Existing prediction-correction methods have been shown to work well for unconstrained convex problems and for settings where obtaining the inverse of the Hessian of the cost function can be computationally affordable. The prediction-correction algorithm proposed in this paper addresses the limitations of existing methods by tackling constrained problems and by designing a first-order prediction step that relies on the Hessian of the cost function. Analytical results are established to quantify the tracking error. Numerical simulations corroborate the analytical results and showcase the performance and benefits of the algorithms.

Keywords: Convex optimization, continuous time system estimation, distributed control and estimation, online algorithms, time-varying optimization.

1. INTRODUCTION

Convex optimization is fundamental to many engineering, societal, and financial problems (Boyd and Vandenberghe, 2004). Core optimization tasks involve numerical algorithms to find the optimal decision variables of a convex program with given inputs and problem parameters. In many applications, one is faced with the formulation (and the solution) of convex programs that *vary continuously over time*. Typical examples include adaptive control, where control actions are generated depending on a (parametric) varying optimization problem (Jerez et al., 2014; Hours and Jones, 2016); and, signal processing problems (Jakubiec and Ribeiro, 2013), where model parameters are estimated online based on streaming observations – including time-varying compressive sensing settings (Angelosante et al., 2010; Yang et al., 2016; Vaswani and Zhan, 2016). Additional application domains include robotics (Verschueren et al., 2009), smart grids (Zhao et al., 2014; Dall’Anese and Simonetto, 2016), and economics (Dontchev et al., 2013).

In this context, we consider the following time-varying constrained optimization problem:

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

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, $f : \mathbb{R}^n \times \mathbb{R}_+ \rightarrow \mathbb{R}$ is a *smooth strongly convex function*. The goal is to find (and track) the solution $\mathbf{x}^*(t)$ of (1) over time – hereafter referred to as the optimal solution *trajectory*. Problem (1) can be conceivably solved based on a continuous time platform (Rahili and Ren, 2016; Fazlyab et al., 2016a,b); alternatively, leveraging sampling arguments, it can be

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

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

By decreasing h , an arbitrary accuracy may be achieved when approximating problem (1) with (2). However, solving (2) in a *batch* fashion for each sampling time t_k may not be computationally affordable in many application domains, even moderately sized problems.

Focusing on unconstrained optimization problems, the works in (Simonetto et al., 2016b,a) developed an online prediction-correction method to find and track the solution trajectory $\mathbf{x}^*(t)$ up to a bounded asymptotical error, starting from an arbitrary guess \mathbf{x}_0 and without solving (2) in a batch setting. The prediction step involves the computation of the inverse of the Hessian of the cost function, and it is utilized to estimate $\mathbf{x}^*(t_{k+1})$, based on information available at time t_k ; subsequently, once the cost function is observed, the correction step fine-tunes the estimate of $\mathbf{x}^*(t_{k+1})$. This methodology is inspired by non-stationary optimization (Polyak, 1987; Popkov, 2005), parametric programming (Robinson, 1980; Dontchev and Rockafellar, 2009; Zavala and Anitescu, 2010; Dontchev et al., 2013), and continuation methods in numerical mathematics (Allgower and Georg, 1990).

Compared to (Simonetto et al., 2016b,a), the contribution of this paper is twofold:

- i) We develop prediction-correction methods to track the solutions of the time-varying *constrained* prob-

lems (1) and provide analytical bounds to characterize their tracking performance;

- 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 (Simonetto et al., 2016b); the proposed prediction-correction method is computationally lighter, as it requires only matrix-vector multiplications. We offer a trade-off between tracking capabilities and computational effort.

In addition, our methods, being first-order, can be directly implemented on networks of computing and communicating nodes, if the problem to be solved is a time-varying networked (i.e., multi-user) optimization problem with local constraints. In this case the methods are *automatically distributed*, without requiring further approximations as required in Simonetto et al. (2016a).

The design and analysis of proposed prediction-correction methods are grounded on the theory of generalized equations and implicit function theorems (Dontchev and Rockafellar, 2009).¹

2. PREDICTION-CORRECTION ALGORITHM

In this section, we focus on problem (1) and design a prediction-correction algorithm to track the (unique) optimal solution trajectory. Pertinent modeling assumptions will be assumed (e.g., the existence and uniform boundedness of the Hessian and higher order derivatives), as explained in Section 3.

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 manner as explained next.

2.1 Prediction

Suppose that \mathbf{x}_k is an approximate solution of (2) at time t_k . Given \mathbf{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 $\mathbf{x}_{k+1|k}$ denote the output of the prediction step.

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

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

¹ **Notation.** Vectors are written as $\mathbf{x} \in \mathbb{R}^n$ and matrices as $\mathbf{A} \in \mathbb{R}^{n \times n}$. 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 $f(\mathbf{x}; t)$ with respect to \mathbf{x} at the point (\mathbf{x}, t) is denoted as $\nabla_{\mathbf{x}} f(\mathbf{x}; t) \in \mathbb{R}^n$, while the partial derivative of the same function with respect to (w.r.t.) t at (\mathbf{x}, t) is written as $\nabla_t f(\mathbf{x}; t) \in \mathbb{R}$. Similarly, the notation $\nabla_{\mathbf{x}\mathbf{x}} f(\mathbf{x}; t) \in \mathbb{R}^{n \times n}$ denotes the Hessian of $f(\mathbf{x}; t)$ w.r.t. \mathbf{x} at (\mathbf{x}, t) , whereas $\nabla_{t\mathbf{x}} f(\mathbf{x}; t) \in \mathbb{R}^n$ denotes the partial derivative of the gradient of $f(\mathbf{x}; t)$ w.r.t. the time t at (\mathbf{x}, t) , i.e. the mixed first-order partial derivative vector of the objective. The tensor $\nabla_{\mathbf{x}\mathbf{x}\mathbf{x}} f(\mathbf{x}; t) \in \mathbb{R}^{n \times n \times n}$ indicates the third derivative of $f(\mathbf{x}; t)$ w.r.t. \mathbf{x} at (\mathbf{x}, t) , the matrix $\nabla_{\mathbf{x}t\mathbf{x}} f(\mathbf{x}; t) = \nabla_{t\mathbf{x}\mathbf{x}} f(\mathbf{x}; t) \in \mathbb{R}^{n \times n}$ indicates the time derivative of the Hessian of $f(\mathbf{x}; t)$ w.r.t. the time t at (\mathbf{x}, t) , and the vector $\nabla_{tt\mathbf{x}} f(\mathbf{x}; t) \in \mathbb{R}^n$ indicates the second derivative in time of the gradient of $f(\mathbf{x}; t)$ w.r.t. the time t at (\mathbf{x}, t) .

where $N_X : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is the normal cone operator, while $\mathbf{x}^*(t_k)$ is the optimizer of (2) at t_k .

In order to compute the predictor $\mathbf{x}_{k+1|k}$, one may wish to solve the generalized equation

$$\nabla_{\mathbf{x}} f(\mathbf{x}_{k+1|k}; t_{k+1}) + N_X(\mathbf{x}_{k+1|k}) \ni \mathbf{0}, \quad (4)$$

yet this is not possible at time t_k . Instead, substitute the gradient $\nabla_{\mathbf{x}} f(\mathbf{x}_{k+1|k}; t_{k+1})$ with a first-order Taylor approximation as:

$$\begin{aligned} \nabla_{\mathbf{x}} f(\mathbf{x}_{k+1|k}; t_{k+1}) &\approx \nabla_{\mathbf{x}} f(\mathbf{x}_k; t_k) + \\ &\nabla_{\mathbf{x}\mathbf{x}} f(\mathbf{x}_k; t_k)(\mathbf{x}_{k+1|k} - \mathbf{x}_k) + h \nabla_{t\mathbf{x}} f(\mathbf{x}_k; t_k), \end{aligned} \quad (5)$$

which is now computable with information available at t_k , and seek the solution of the following perturbed generalized equation

$$\begin{aligned} \nabla_{\mathbf{x}} f(\mathbf{x}_k; t_k) + \nabla_{\mathbf{x}\mathbf{x}} f(\mathbf{x}_k; t_k)(\mathbf{x}_{k+1|k} - \mathbf{x}_k) + \\ h \nabla_{t\mathbf{x}} f(\mathbf{x}_k; t_k) + N_X(\mathbf{x}_{k+1|k}) \ni \mathbf{0}. \end{aligned} \quad (6)$$

That is, the prediction step produces a solution that is optimal w.r.t. a perturbed (first-order) version of the original generalized equation (4). We can now replace (6) with the following equivalent formulation

$$\begin{aligned} \mathbf{x}_{k+1|k} = \operatorname{argmin}_{\mathbf{x} \in X} \left\{ \frac{1}{2} \mathbf{x}^\top \nabla_{\mathbf{x}\mathbf{x}} f(\mathbf{x}_k; t_k) \mathbf{x} + (\nabla_{\mathbf{x}} f(\mathbf{x}_k; t_k) \right. \\ \left. + h \nabla_{t\mathbf{x}} f(\mathbf{x}_k; t_k) - \nabla_{\mathbf{x}\mathbf{x}} f(\mathbf{x}_k; t_k) \mathbf{x}_k)^\top \mathbf{x} \right\}. \end{aligned} \quad (7)$$

Problem (7) is a constrained optimization problem with quadratic cost, and unless the dimension of the problem is small, its solution may be too computationally demanding to be implemented on tight time requirements (as will be shown in the Numerical Results section). Therefore, we consider the less computationally demanding task of finding an approximate solution of (7) by computing a number of projected gradient descent steps – the first key step towards a *first-order* prediction-correction method. Particularly, let $\hat{\mathbf{x}}^0$ be a dummy variable initialized as $\hat{\mathbf{x}}^0 = \mathbf{x}_k$; then, the following steps are performed:

$$\begin{aligned} \hat{\mathbf{x}}^{p+1} = \mathbb{P}_X[\hat{\mathbf{x}}^p - \alpha(\nabla_{\mathbf{x}\mathbf{x}} f(\mathbf{x}_k; t_k)(\hat{\mathbf{x}}^p - \mathbf{x}_k) + \\ h \nabla_{t\mathbf{x}} f(\mathbf{x}_k; t_k) + \nabla_{\mathbf{x}} f(\mathbf{x}_k; t_k))], \end{aligned} \quad (8)$$

for $p = 0, 1, \dots, P-1$, where P is a predetermined 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, $\tilde{\mathbf{x}}_{k+1|k}$ is set to:

$$\tilde{\mathbf{x}}_{k+1|k} = \hat{\mathbf{x}}^P. \quad (9)$$

2.2 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 $\mathbf{x}^*(t_{k+1})$. To this end, a first-order projected gradient method is considered next. Particularly, let $\hat{\mathbf{x}}^0 = \tilde{\mathbf{x}}_{k+1|k}$ be a dummy variable; then, consider the following projected gradient steps

$$\hat{\mathbf{x}}^{c+1} = \mathbb{P}_X[\hat{\mathbf{x}}^c - \beta(\nabla_{\mathbf{x}} f(\hat{\mathbf{x}}^c; t_{k+1}))], \quad (10)$$

for $c = 0, 1, \dots, C-1$, where C is a predetermined number of gradient steps and $\beta > 0$ the stepsize. The estimate of the optimal solution $\mathbf{x}^*(t_{k+1})$ is then computed as $\mathbf{x}_{k+1} = \hat{\mathbf{x}}^C$. Notice that a Newton step could be implemented; however, to develop computationally light online schemes that naturally afford a distributed implementation, this paper considers first-order methods.

Algorithm 1
Constrained First-Order Prediction-Correction (C-FOPC)
Require: Initial variable \mathbf{x}_0 . Initial objective function $f(\mathbf{x}; t_0)$, no. of prediction steps P and correction steps C

```

1: for  $k = 0, 1, 2, \dots$  do
2:   // time  $t_k$ 
3:   Prediction: initialize  $\hat{\mathbf{x}}^0 = \mathbf{x}_k$ 
4:   for  $p = 0 : P - 1$  do
5:     Predict the variable by the gradient step [cf (8)]
            $\hat{\mathbf{x}}^{p+1} = \mathbb{P}_X[\hat{\mathbf{x}}^p - \alpha(\nabla_{\mathbf{x}\mathbf{x}}f(\mathbf{x}_k; t_k)(\hat{\mathbf{x}}^p - \mathbf{x}_k) +$ 
            $h \nabla_{t\mathbf{x}}f(\mathbf{x}_k; t_k) + \nabla_{\mathbf{x}}f(\mathbf{x}_k; t_k))]$ 
6:   end for
7:   Set the predicted variable  $\tilde{\mathbf{x}}_{k+1|k} = \hat{\mathbf{x}}^P$ 
8:   // time  $t_{k+1}$ 
9:   Acquire the updated function  $f(\mathbf{x}; t_{k+1})$ 
10:  Initialize the sequence of corrected variables  $\hat{\mathbf{x}}^0 = \tilde{\mathbf{x}}_{k+1|k}$ 
11:  for  $c = 0 : C - 1$  do
12:    Correct the variable by the gradient step [cf (10)]
            $\hat{\mathbf{x}}^{c+1} = \mathbb{P}_X[\hat{\mathbf{x}}^c - \beta(\nabla_{\mathbf{x}}f(\hat{\mathbf{x}}^c; t_{k+1}))]$ 
13:  end for
14:  Set the corrected variable  $\mathbf{x}_{k+1} = \hat{\mathbf{x}}^C$ 
15: end for

```

2.3 Complete Algorithm

The complete algorithm Constrained - First-Order Prediction Correction (C-FOPC) is tabulated as Algorithm 1. Steps 4-7 are utilized to compute $\tilde{\mathbf{x}}_{k+1|k}$ based on the information available at t_k . Provided that the projection operator is easy to carry out (set X is simple), 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 (Simonetto et al., 2016b), which involve the computation of the Hessian inverse. Steps 10-14 are utilized to compute \mathbf{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.

In Figure 1, we offer a pictorial representation of one step of the exact prediction-correction methodology. This picture is displayed to offer some intuition on the algorithm and on what the first-order strategy tries to approximate. The time-varying problem is a constrained (over the set X) problem in two dimensions x_1 and x_2 . The function varies over time by moving rigidly on the plane. Two time instances t_k and t_{k+1} are represented, while we see in orange the continuous optimal trajectory traversing the constraint set. At time step t_k , we are given the approximate optimizer $\mathbf{x}_k \in X$. We then predict the new approximate optimizer $\mathbf{x}_{k+1|k}$. For simplicity, imagine that the constraint set is not active, so that we can solve (7) exactly as

$$\mathbf{x}_{k+1|k} = \mathbf{x}_k - \underbrace{h \left[\nabla_{\mathbf{x}\mathbf{x}}f(\mathbf{x}_k; t_k) \right]^{-1} \nabla_{t\mathbf{x}}f(\mathbf{x}_k; t_k)}_{(1)} - \underbrace{\left[\nabla_{\mathbf{x}\mathbf{x}}f(\mathbf{x}_k; t_k) \right]^{-1} \nabla_{\mathbf{x}}f(\mathbf{x}_k; t_k)}_{(2)}. \quad (11)$$

Therefore, we compute $\mathbf{x}_{k+1|k}$ by combining two terms: term (1) is towards points that have similar gradient

$\nabla_{\mathbf{x}}f(\mathbf{x}_k; t_k)$ at time t_{k+1} , while term (2) is a Newton term towards the optimizer at t_k . The first-order approximated strategy that we employ in Algorithm 1 attempts to mimic this behavior, moving both in the direction of equal gradients at successive times (i.e., following the movement of the function) and in the direction of reduced suboptimality at time t_k . Lastly, the correction step moves towards the optimizer at t_{k+1} .

Remark 1. [Time-derivative approximation] The time derivative of the gradient $\nabla_{t\mathbf{x}}f(\mathbf{x}; t)$ can be substituted with an approximate version, as explained in (Simonetto et al., 2016b).

Remark 2. [Unconstrained problems] The presented algorithm works for unconstrained problems too, by substituting $X = \mathbb{R}^n$. In this case, Eq. (11) is an improvement compared to the prediction step of (Simonetto et al., 2016b) (featuring only term (1)). A detailed discussion is available in (Simonetto and E. Dall’Anese, 2016).

3. CONVERGENCE ANALYSIS

In this section, we establish analytical results to bound the discrepancy between the optimal solution $\mathbf{x}^*(t)$ and the iterates \mathbf{x}_k produced by the prediction-correction scheme presented in the previous section. Some technical conditions are required as stated next

Assumption 1. The function $f(\mathbf{x}; t)$ is twice differentiable and m -strongly convex in $\mathbf{x} \in X$ and uniformly in t ; that is, the Hessian of $f(\mathbf{x}; t)$ with respect to \mathbf{x} is bounded below by m for each $\mathbf{x} \in X$ and uniformly in t ,

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

Assumption 2. The function $f(\mathbf{x}; t)$ is sufficiently smooth both in $\mathbf{x} \in X$ and in t ; in particular, $f(\mathbf{x}; t)$ has bounded second and third-order derivatives with respect to $\mathbf{x} \in X$ and t :

$$\begin{aligned} \|\nabla_{\mathbf{x}\mathbf{x}}f(\mathbf{x}; t)\| &\leq L, \quad \|\nabla_{t\mathbf{x}}f(\mathbf{x}; t)\| \leq C_0, \quad \|\nabla_{\mathbf{x}\mathbf{x}\mathbf{x}}f(\mathbf{x}; t)\| \leq C_1, \\ \|\nabla_{\mathbf{x}t\mathbf{x}}f(\mathbf{x}; t)\| &\leq C_2, \quad \|\nabla_{tt\mathbf{x}}f(\mathbf{x}; t)\| \leq C_3. \end{aligned}$$

Assumption 1 guarantees that problem (1) is strongly convex and has a *unique* solution for each time instance. On course, uniqueness of the solution implies that the solution trajectory is also unique. This setting is common in the the time-varying optimization domain; see, for instance (Popkov, 2005; Dontchev et al., 2013; Jakubiec and Ribeiro, 2013; Ling and Ribeiro, 2014; Simonetto et al., 2016b,a). Assumption 2 ensures that the Hessian is bounded from above; this property is equivalent to the Lipschitz continuity of the gradient; it also ensures that the third derivative tensor $\nabla_{\mathbf{x}\mathbf{x}\mathbf{x}}f(\mathbf{x}; t)$ is bounded above (typically required for the analysis of Newton-type algorithms), as well as 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.

Assumptions 1 and 2 are sufficient to show that the solution *mapping* $t \mapsto \mathbf{x}^*(t)$ is single-valued and locally Lipschitz continuous in t ; in particular, from (Dontchev and Rockafellar, 2009, Theorem 2F.10) we have that

$$\|\mathbf{x}^*(t_{k+1}) - \mathbf{x}^*(t_k)\| \leq \frac{1}{m} \|\nabla_{t\mathbf{x}}f(\mathbf{x}; t)\|(t_{k+1} - t_k) \leq \frac{C_0 h}{m}, \quad (12)$$

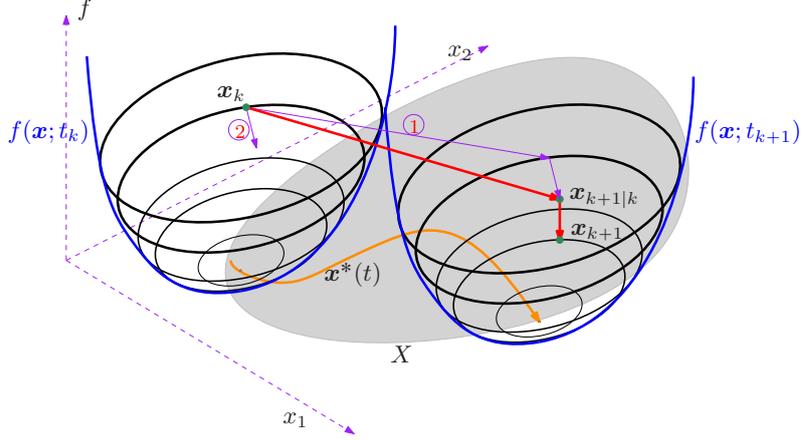


Fig. 1. Pictorial representation of one step of the exact prediction-correction strategy applied on a moving quadratic cost function.

for sufficiently small sampling periods h . This result established link between the sampling period h and the temporal variability of the optimal solutions; further, (12) will be utilized to substantiate the convergence and tracking capabilities of the proposed prediction-correction methods.

Eq. (12) provides a link between the sampling period h and the allowed variations in the optimizers. This also gives a better understanding on the time-varying assumptions on the uniform boundedness of the time derivatives of the gradient $\nabla_{t\mathbf{x}}f(\mathbf{x}; t)$ and $\nabla_{tt\mathbf{x}}f(\mathbf{x}; t)$. In fact, the bounds C_0 and C_3 require that the change and the rate of change of the optimizer be bounded. If the optimizer were the position of a moving target to be estimated, then C_0 and C_3 would be a bound on its velocity and acceleration. Finally, the bound on $\nabla_{\mathbf{x}t\mathbf{x}}f(\mathbf{x}; t)$ means that the quantity $\nabla_{t\mathbf{x}}f(\mathbf{x}; t)$ is Lipschitz continuous w.r.t. \mathbf{x} uniformly in t ; i.e., that close by points \mathbf{x} and \mathbf{x}' need to have similar gradient time-derivatives: e.g., if the target position is perturbed by a small amount $\delta\mathbf{x}$ then its velocity is perturbed by an amount not bigger than $C_2\delta\mathbf{x}$.

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

Theorem 1. Consider the sequence $\{\mathbf{x}_k\}_{k \in \mathbb{N}}$ generated by C-FOPC, and let Assumptions 1-2 hold true. Define the following quantities

$$\varrho_P = \max\{|1-\alpha m|, |1-\alpha L|\}, \varrho_C = \max\{|1-\beta m|, |1-\beta L|\}, \quad (13)$$

and let stepsizes α and β be such that

$$\alpha < 2/L, \quad \beta < 2/L. \quad (14)$$

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 $h \leq \bar{h}$ and the initial optimality gap satisfy $\|\mathbf{x}_0 - \mathbf{x}^*(t_0)\| \leq \bar{R}$, then the sequence $\{\|\mathbf{x}_k - \mathbf{x}^*(t_k)\|\}_{k \in \mathbb{N}}$ converges linearly with rate τ to an asymptotical error bound, and

$$\limsup_{k \rightarrow \infty} \|\mathbf{x}_k - \mathbf{x}^*(t_k)\| = O(h^2 \varrho_C^C) + O(h \varrho_P^P \varrho_C^C). \quad (15)$$

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

$$\bar{h} = \frac{\tau - \varrho_C^C \varrho_P^P}{\varrho_C^C (\varrho_P^P + 1)} \left(\frac{C_1 C_0}{m^2} + \frac{C_2}{m} \right)^{-1}, \bar{R} = \frac{2m}{C_1} \left(\frac{C_1 C_0}{m^2} + \frac{C_2}{m} \right) (\bar{h} - h). \quad (16)$$

Proof. See (Simonetto and E. Dall'Anese, 2016), where we also characterize the constants in the right-hand side of (15).

Theorem 1 asserts that the sequence $\{\mathbf{x}_k\}_{k \in \mathbb{N}}$ generated by C-FOPC locks to a neighborhood of the optimal solution trajectory $\mathbf{x}^*(t)$. 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 $P \rightarrow \infty$), then the bound goes as $O(h^2)$, which is similar to the bounds derived in (Simonetto et al., 2016b). When one performs the correction step exactly, i.e., $C \rightarrow \infty$, then the asymptotic bound goes to zero (in fact, in that case each the time-invariant problem is solved exactly).

The presence of an attraction region is due to Newton steps in the prediction stage (that is, the presence of the gradient $\nabla_{\mathbf{x}}f(\mathbf{x}_k; t_k)$ in the generalized equation (6)). When the function is quadratic, then $C_1 = 0$, and the convergence is global.

4. NUMERICAL RESULTS

To appreciate the merits of the proposed algorithm, consider an optimization problem with $n = 1000$ (scalar) variables. The cost function we study has the form

$$f(\mathbf{x}; t) = \frac{1}{2} \|\mathbf{x} + \mathbf{1}_n\|_{\mathbf{Q}}^2 + \sum_{i=1}^n \kappa_i \sin^2(\omega t + \varphi_i) \exp(\mu(x_{(i)} - 2)^2), \quad (17)$$

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 $\mathbf{x} \in \mathbb{R}^n$. In addition, the matrix \mathbf{Q} is chosen as $\mathbf{Q} = \mathbf{I}_n + \boldsymbol{\mu}\boldsymbol{\mu}^\top/n$ 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

$$\underset{\mathbf{x} \in [0, 0.4]^n}{\text{minimize}} f(\mathbf{x}; t). \quad (18)$$

We notice that the cost function f verifies the Assumptions 1-2 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 $L = 6.07$.

We focus our performance analysis on realistic run-time constraints and we compare our prediction-correction strategy with state-of-the-art correction-only methodologies (i.e., no prediction step is performed) [See Simonetto and E. Dall’Anese (2016) for a detailed discussion on these methodologies].

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

$$C = \lfloor r_1 h / t_C \rfloor, \quad (19)$$

correction steps. After the corrected variable is available, one can use it for the decision making process, e.g., to determine the control law, 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 $\bar{t} + P t_P$, 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,

$$P = \lfloor (r_2 h - \bar{t}) / t_P \rfloor. \quad (20)$$

The affordable extra correction steps C' can be computed as in (19), 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 $t_C = .76$ ms, $\bar{t} = 10$ ms, $t_P = .62$ 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

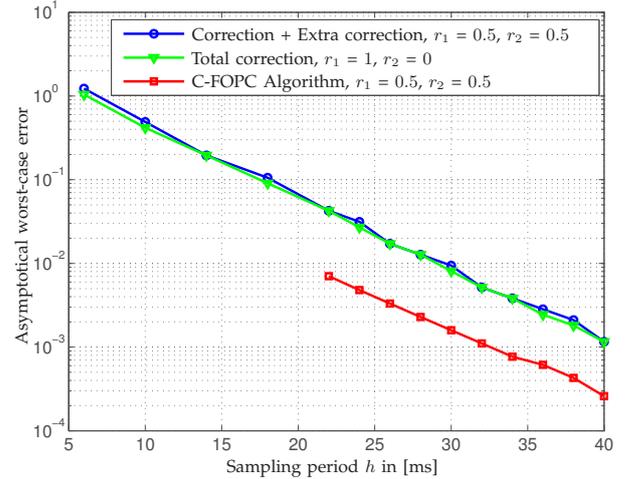


Fig. 2. Asymptotical worst case error floor with respect to the sampling time interval h for different algorithms applied to (18).

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 2, 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), while the number of prediction steps and correction steps are optimized via the available resources as in Eq.s (19)-(20). 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 $h = [22, 40]$ ms, then $\bar{h} = [90, 370]$ ms, and $\bar{R} = [.13, .68]$. In this simulation example, if the prediction is affordable, we see clearly that the prediction-correction algorithm?our C-FOPC algorithm?is preferred instead of the correction-only schemes because 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 \mathbf{x}_0 is chosen to be zero, while the initial optimality gap is .30, which indicates that our bounds are somewhat conservative.

Surprisingly, the result suggests 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 3, we report the time trajectories of a number of variables for the three strategies to appreciate how the constraints are, in fact, active.

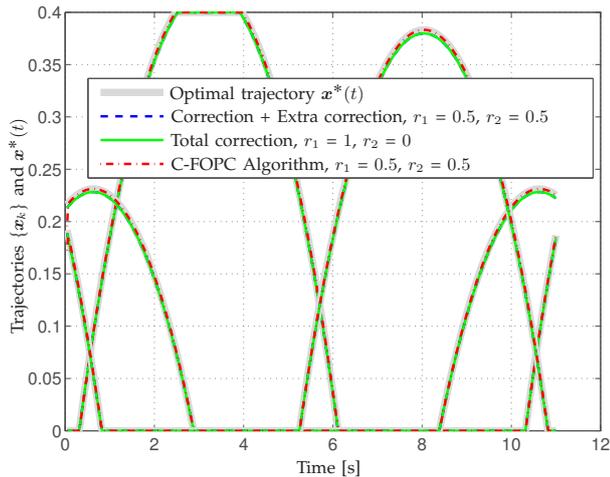


Fig. 3. Trajectories of $\{x_k\}$ and optimal trajectory $x^*(t)$ for different algorithms applied to (18), with $h = 22$ ms.

5. CONCLUSION

We have presented first-order algorithms to find and track the solution trajectory of a class of smooth, strongly convex constrained time-varying optimization problems. These algorithms rely on a prediction-correction mechanism, which exhibit better asymptotical accuracy than state-of-the-art correction-only schemes, even when computational complexity issues are taken into account.

REFERENCES

- Allgower, E.L. and Georg, K. (1990). *Numerical Continuation Methods: An Introduction*. Springer-Verlag.
- Angelosante, D., Bazerque, J.A., and Giannakis, G.B. (2010). Online adaptive estimation of sparse signals: Where rls meets the ℓ_1 -norm. *IEEE Trans. on Signal Processing*, 58(7), 3436–3447.
- Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.
- Dall’Anese, E. and Simonetto, A. (2016). Optimal Power Flow Pursuit. *IEEE Transactions on Smart Grid*, in press.
- Dontchev, A.L., Krastanov, M.I., Rockafellar, R.T., and Veliov, V.M. (2013). An Euler-Newton Continuation method for Tracking Solution Trajectories of Parametric Variational Inequalities. *SIAM Journal of Control and Optimization*, 51(51), 1823 – 1840.
- Dontchev, A.L. and Rockafellar, R.T. (2009). *Implicit Functions and Solution Mappings*. Springer.
- Fazlyab, M., Paternain, S., Preciado, V., and Ribeiro, A. (2016a). Interior Point Method for Dynamic Constrained Optimization in Continuous Time. In *Proceedings of the American Control Conference*, 5612 – 5618. Boston (MA), USA.
- Fazlyab, M., Paternain, S., Preciado, V., and Ribeiro, A. (2016b). Prediction-Correction Interior-Point Method for Time-Varying Convex Optimization. Arxiv: 1608.07544.
- Hours, J.H. and Jones, C.N. (2016). A Parametric Non-Convex Decomposition Algorithm for Real-Time and Distributed NMPC. *IEEE Transactions on Automatic Control*, 61(2), 287 – 302.
- Jakubiec, F.Y. and Ribeiro, A. (2013). D-MAP: Distributed Maximum a Posteriori Probability Estimation of Dynamic Systems. *IEEE Transactions on Signal Processing*, 61(2), 450 – 466.
- Jerez, J.L., Goulart, P.J., Richter, S., Constantinides, G.A., Kerrigan, E.C., and Morari, M. (2014). Embedded Online Optimization for Model Predictive Control at Megahertz Rates. *IEEE Transactions on Automatic Control*, 59(12), 3238 – 3251.
- Ling, Q. and Ribeiro, A. (2014). Decentralized Dynamic Optimization Through the Alternating Direction Method of Multipliers. *IEEE Transactions on Signal Processing*, 62(5), 1185 – 1197.
- Polyak, B.T. (1987). *Introduction to Optimization*. Optimization Software, Inc.
- Popkov, A.Y. (2005). Gradient Methods for Nonstationary Unconstrained Optimization Problems. *Automation and Remote Control*, 66(6), 883 – 891. Translated from *Avtomatika i Telemekhanika*, No. 6, 2005, pp. 38 – 46.
- Rahili, S. and Ren, W. (2016). Distributed Convex Optimization for Continuous-Time Dynamics with Time-Varying Cost Functions. *IEEE Transactions on Automatic Control (to appear)*.
- Robinson, S.M. (1980). Strongly Regular Generalized Equations. *Mathematics of Operations Research*, 5(1), 43 – 62.
- Simonetto, A. and E. Dall’Anese (2016). Prediction-Correction Algorithms for Time-Varying Constrained Optimization. Available on arXiv.
- Simonetto, A., Koppel, A., Mokhtari, A., Leus, G., and Ribeiro, A. (2016a). Decentralized Prediction-Correction Methods for Networked Time-Varying Convex Optimization. arXiv: 1602.01716.
- Simonetto, A., Mokhtari, A., Koppel, A., Leus, G., and Ribeiro, A. (2016b). A Class of Prediction-Correction Methods for Time-Varying Convex Optimization. *IEEE Transactions on Signal Processing*, 64(17), 4576 – 4591.
- Vaswani, N. and Zhan, J. (2016). Recursive Recovery of Sparse Signal Sequences from Compressive Measurements: A Review. *IEEE Transactions on Signal Processing*, 64(13), 3523 – 3549.
- Verscheure, D., Demeulenaere, B., Swevers, J., De Schutter, J., and Diehl, M. (2009). Time-Optimal Path Tracking for Robots: a Convex Optimization Approach. *IEEE Transactions on Automatic Control*, 54(10), 2318 – 2327.
- Yang, Y., Zhang, M., Pesavento, M., and Palomar, D.P. (2016). An Online Parallel and Distributed Algorithm for Recursive Estimation of Sparse Signals. *IEEE Transactions on Signal and Information Processing over Networks*, 2(3), 290 – 305.
- Zavala, V.M. and Anitescu, M. (2010). Real-Time Non-linear Optimization as a Generalized Equation. *SIAM Journal of Control and Optimization*, 48(8), 5444 – 5467.
- Zhao, C., Topcu, U., Li, N., and Low, S. (2014). Design and Stability of Load-Side Primary Frequency Control in Power Systems. *IEEE Transactions on Automatic Control*, 59(5), 1177 – 1189.