A Decentralized Prediction-Correction Method for Networked Time-Varying Convex Optimization

Andrea Simonetto§, Aryan Mokhtari*, Alec Koppel*, Geert Leus§, and Alejandro Ribeiro*

Dept. of EEMCS, Delft University of Technology, 2826 CD Delft, The Netherlands
{a.simonetto, g.j.t.leus}@tudelft.nl
*Dept. of ESE, University of Pennsylvania, 200 South 33rd Street, Philadelphia, PA 19104, USA
{akoppel, ariyann, aribeiro}@seas.upenn.edu

Abstract—We study networked unconstrained convex optimization problems where the objective function changes continuously in time. We propose a decentralized algorithm (DePCoT) with a discrete time-sampling scheme to find and track the solution trajectory based on prediction and gradient-based correction steps, while sampling the problem data at a constant sampling period $h$. Under suitable conditions and for limited sampling periods, we establish that the asymptotic error bound behaves as $O(h^2)$, which outperforms the state of the art existing error bound of $O(h)$ for correction-only methods. The key contributions are the prediction step and a decentralized method to approximate the inverse of the Hessian of the cost function in a decentralized way, which yields quantifiable trade-offs between communication and accuracy.

I. INTRODUCTION

In this paper, we consider convex optimization problems where the objective function changes in time and its components are available at different nodes of a network. The objective function can be decomposed into two parts where the first part is locally available at nodes and the second part is shared between neighboring nodes. To be more precise, consider a connected undirected network containing $n$ nodes where $y^i \in \mathbb{R}^p$ is the decision variable of node $i$. Define $y = [y^1; \ldots; y^n] \in \mathbb{R}^{np}$ as the concatenation of the decision variables. Nodes aim at cooperatively minimizing the global cost function $F: \mathbb{R}^{np} \times \mathbb{R}_+ \rightarrow \mathbb{R}$, which can be written as the sum of a locally available function $\phi_i: \mathbb{R}^p \times \mathbb{R}_+ \rightarrow \mathbb{R}$ and a network related function $G: \mathbb{R}^{np} \times \mathbb{R}_+ \rightarrow \mathbb{R}$. Therefore, the optimization problem is

$$\arg\min_{y \in \mathbb{R}^{np}} F(y; t) := \arg\min_{y \in \mathbb{R}^p} \phi_i(y; t) + G(y; t).$$

(1)

Notice that nodes can minimize the objective function $\phi_i(y; t)$ independently, while minimizing of the objective function $G(y; t)$ requires coordination and exchanging information between neighboring nodes. Problems of form (1) arise, e.g., in time-varying versions of multiuser network optimization and resource allocation, see for example [1], [2] for time-invariant distributed algorithms for these problems.

We consider using and extending the tools of non-stationary optimization [3]–[5] to solve problems of the form (1) by prediction-correction algorithms. We start the paper we introducing an equivalent formulation of (1) which is more suitable for decentralized optimization and define the discretized version of (1) (Section II) by sampling it at a constant rate $1/h$. Then, we study a brief discussion of Gradient Trajectory Tracking (GTT) algorithm which uses a prediction-correction scheme for minimizing dynamic optimization problems in centralized settings (Section II-A). GTT predicts the optimal solution at the discrete time instance $t_{k+1}$ by approximating variation of the objective function $F$ from $t_k$ to $t_{k+1}$ and corrects the predicted solution by executing a single step of projected gradient descent. However, GTT is not applicable to decentralized optimization problems since the prediction step requires access to Hessian inverse of the objective function $F$ which is not computable in a decentralized manner. We propose a Decentralized Prediction-Correction Tracking (DePCoT) algorithm that approximates the prediction direction of GTT by truncating the Taylor series of the objective function Hessian inverse (Section II-B). We show a trade-off in the implementation of DePCoT between approximation accuracy and communication cost. We follow the paper by analyzing convergence properties of DePCoT (Section III). We prove that under some specific conditions and for limited sampling periods $h$, the sequence of iterates generated by DePCoT converges linearly to a tracking error of $O(h^2)$ (Theorem 1). This result improves the error of state-of-the art decentralized correction-only methods (so-called running methods) [6]–[8] which is in the order of $O(h)$. Finally, we present numerical simulations to display the added value of DePCoT in an estimation problem of a spatially distributed process. The proofs of the results are available in [9].

II. PROBLEM FORMULATION AND ALGORITHM DEFINITION

Consider a connected undirected network containing $n$ nodes and define $N^i$ as the neighborhood of node $i$, i.e., the nodes it can exchange information with. Consider the vector $y^i \in \mathbb{R}^p$ as the decision variable of node $i$. Each node $i$ has access to a time-varying local function $\phi_i^j: \mathbb{R}^p \times \mathbb{R}_+ \rightarrow \mathbb{R}$ where its input arguments are the local variable $y^i$ and time $t$. Further, define $g^{i,j}: \mathbb{R}^p \times \mathbb{R}_+ \rightarrow \mathbb{R}$ as a common objective function between nodes $i$ and $j$ which takes $y^i, y^j,$ and $t$ as inputs. The shared functions are symmetric $g^{i,j}(y^i, y^j; t) = g^{j,i}(y^j, y^i; t)$. Nodes aim at minimizing their local objective function $\phi^i$ while they collaborate with their neighbors to minimize the shared functions $g^{i,j}$: i.e., nodes aim at cooperatively solving the problem

$$\min_{y^1, \ldots, y^n} \sum_{i=1}^n \left( \phi^i(y^i; t) + g^{i,j}(y^i, y^j; t) + \sum_{j \in N^i} g^{j,i}(y^j, y^i; t) \right).$$

(2)

Let $y = [y^1; \ldots; y^n] \in \mathbb{R}^{np}$ be the stacked vector containing all the local decision variables as in (1). If we define the time-varying objective functions

$$\phi(y; t) := \sum_{i=1}^n \phi^i(y^i; t), \quad G(y; t) := \sum_{i=1}^n \sum_{j \in N^i} g^{i,j}(y^i, y^j; t),$$

(3)

the optimization problem in (2) can be written as (1).

To solve the dynamic optimization problem in (2) or its equivalent (1), the first step is sampling the objective function $F$ at time instants $t_k$ with $k = 0, 1, 2, \ldots$ which leads to the sequence of time-invariant problems

$$y^i(t_k) := \arg\min_{y^i \in \mathbb{R}^p} F(y^i, t_k) \quad k \geq 0.$$

(4)

This research was supported in part by STW under the D2S2 project from the ASSYS program (project 10561) and in part by NSF CAREER CCF-0952867, and ONR N00014-12-1-0997.
This give rise to a discrete sequence \( \{y_k\} \) that needs to be made close to the optimal argument trajectory \( y^*(t_k) \). In the following section we study the Gradient Trajectory Tracking (GTT) algorithm as a centralized method for solving the sequence of optimization problems in (4).

A. Gradient Trajectory Tracking

The GTT method executes a prediction-correction scheme to first estimate the change of optimal arguments from \( t_{k-1} \) to \( t_k \) and then correct the predicted solution by running a step of gradient descent. The prediction step is built on modeling evolution of the trajectory \( y(t) \) with a defined residual: For each \( y(t) \), we can write \( \nabla_y F(y(t); r(t)) \), where \( r(t) \) is the residual vector (it is zero only at optimality). By perturbing this gradient condition for variations \( \delta t \) and \( \delta y \), we arrive at the dynamic system [10]

\[
\dot{y} = -[\nabla_{yy} F(y; t)]^{-1}\nabla_{ty} F(y; t),
\]

where \( \nabla_{ty} F(y; t) \in \mathbb{R}^{np} \) and \( \nabla_{yy} F(y; t) \in \mathbb{R}^{np \times np} \) are the mixed partial derivative and Hessian of the objective function \( F \), respectively. By sampling at sampling times \( t_k \), for \( k = 0, 1, 2, \ldots \) using a first-order forward Euler scheme for the relation in (5), the predicted variable \( y_{k+1|k} \) is given by

\[
y_{k+1|k} = y_k - h [\nabla_{yy} F(y_k; t_k)]^{-1}\nabla_{ty} F(y_k; t_k).
\]

The predicted variable \( y_{k+1|k} \) computed as in (6) is corrected by a step of projected gradient descent with stepsize \( \gamma > 0 \)

\[
y_{k+1} = P_\gamma \left[ y_{k+1|k} - \gamma \nabla_y F(y_{k+1|k}; t_k+1) \right].
\]

Based on the definition of the objective function \( F \) in (1), the Hessian inverse \( \nabla_{yy}^2 F(y_k; t_k) \) can be written as

\[
\nabla_{yy} F(y_k; t_k) = \nabla_{yy}^2 F(y_k; t_k) + \nabla_{yy} G(y_k; t_k),
\]

where \( \nabla_{yy}^2 F(y_k; t_k) \in \mathbb{R}^{np \times np} \) is a block diagonal matrix formed by the Hessian of local functions \( \phi \). In other words, the i-th diagonal block of \( \nabla_{yy}^2 F(y_k; t_k) \) is given by \( \nabla_y \phi_i(y_k; t_k) \). Further, \( \nabla_{yy} G(y_k; t_k) \in \mathbb{R}^{np \times np} \) has the sparsity pattern of the graph, since its i-j-th block \( \nabla_{yy} G(y_k; t_k)_{ij} \in \mathbb{R}^{np \times np} \) is not null if and only if \( j = i \) or \( j \notin N_i \). Combining the two matrices we obtain that the Hessian inverse \( \nabla_{yy}^2 F(y_k; t_k) \) has the sparsity pattern of the graph, therefore, it can be computed in a decentralized manner. Although, the objective function \( H_k \) is not necessarily sparse, we can approximate the time varying Hessian \( \nabla_{yy}^2 F(y_k; t_k) \) by a graph sparse matrix.

B. Decentralized Prediction-Correction tracking

To implement the prediction step in (6), the Hessian inverse \( [\nabla_{yy}^2 F(y_k; t_k)]^{-1} \) is required, however, it is not necessarily graph sparse. This means that it cannot be computed only by 1-hop communication. To overcome this difficulty, we leverage and generalize a recently proposed distributed algorithm to approximate Hessian inverses up to an arbitrary accuracy [11], [12]. The approximations are obtained by truncation of the Hessian inverse Taylor expansion. To be more precise, let \( H_k \) be the objective function Hessian \( \nabla_{yy}^2 F(y_k; t_k) \) computed at time \( t_k \) for \( y_k \), i.e., \( H_k = \nabla_{yy}^2 F(y_k; t_k) \). Further, define \( L \) as the largest eigenvalue of the positive semi-definite matrix \( \nabla_{yy}^2 G(y_k; t_k) \) [cfr. Assumption 3]. We write the Hessian as \( H_k := D_k - B_k \), where matrices \( D_k \) and \( B_k \) are defined as

\[
D_k := \nabla_{yy} F(y_k; t_k) + L I, \quad B_k := L I - \nabla_{yy} G(y_k; t_k).
\]

By assuming strong convexity of the function \( \Phi \) [cfr. Assumption 2], the matrix \( D_k \) is a positive definite block diagonal matrix and encodes the local effects; the matrix \( B_k \) is positive semi-definite by construction and has the same structure of the graph. By definition \( H_k := D_k - B_k \), given that \( D_k \) is a positive definite block diagonal matrix, the objective function Hessian \( H_k \) can be written as \( H_k = D_k^{-1/2} (I - D_k^{-1/2} B_k D_k^{-1/2}) D_k^{1/2} \). To compute the Hessian inverse \( H_k^{-1} \) we can use the Taylor series\(^1 \)

\[
(I - X)^{-1} = \sum_{\tau=0}^{\infty} X^\tau \text{ for } X = D_k^{-1/2} B_k D_k^{-1/2}
\]

to obtain

\[
H_k^{-1} = D_k^{-1/2} \sum_{\tau=0}^{\infty} \left(D_k^{-1/2} B_k D_k^{-1/2}\right)^\tau D_k^{-1/2}.
\]

We introduce the Decentralized Prediction-Correction Tracking (DePCoT) as a decentralized algorithm that approximates the Hessian inverse \( H_k^{-1} \) in (6) by truncating the series in (10). The approximate Hessian inverse \( \tilde{H}_k^{-1}(k) \) for DePCoT with \( K \) level of approximation is defined by the first \( K+1 \) terms in (10) as

\[
\tilde{H}_k^{-1}(k) = D_k^{-1/2} \sum_{\tau=0}^{K} \left(D_k^{-1/2} B_k D_k^{-1/2}\right)^\tau D_k^{-1/2}.
\]

The Hessian inverse approximation in (11) follows that the prediction step of DePCoT can be written as

\[
y_{k+1|k} = y_k - h \tilde{H}_k^{-1}(k) \nabla_y F(y_k; t_k) := y_k - h d_{k}(k),
\]

where \( d_{k}(k) := \tilde{H}_k^{-1}(k) \nabla_y F(y_k; t_k) \) is defined as the prediction direction of DePCoT for \( K \) level of approximation.

The predicted variable \( y_{k+1|k} \) of DePCoT at step \( k+1 \) is corrected by descending through the negative objective function gradient \( \nabla_y F(y_{k+1|k}; t_{k+1}) \in \mathbb{R}^{np} \). Therefore, the correction step of DePCoT is identical to (7) and given by

\[
y_{k+1} = P_Y \left[ y_{k+1|k} - \gamma \nabla_y F(y_{k+1|k}; t_{k+1}) \right],
\]

where \( \gamma > 0 \) is the stepsize and \( P_Y \) is the projection operator to the convex set \( Y = Y^1 \times \cdots Y^n \) [cfr. Assumption 1].

The prediction and correction steps of DePCoT in (7) and (13), respectively, are implementable in a decentralized manner. To study this statement define the components \( d_i^{(k)}(k) \in \mathbb{R}^{np} \) of DePCoT’s prediction direction \( d_{k}(k) := [d_{1}^{(k)}(k); \ldots d_{p}^{(k)}(k)] \in \mathbb{R}^{np} \). The important observation is that node \( i \) can compute its prediction direction \( d_{k}(k) \) by exchanging information with its neighbors. To be more precise, the sequence of DePCoT prediction directions satisfies [9]

\[
d_{i}(k) = D_k^{-1} (B_k d_{i-1}(k) - \nabla_{ty} F(y_k; t_k))
\]

Considering the graph sparse structure of \( B_k \) and block diagonality of \( D_k \), the local components of the prediction directions are related to each other as

\[
d_i^{(k)}(k) = (D_k^{(i)})^{-1} \left( \sum_{j \in N_i} B_{ij}^{(i)} d_j^{(i-1)}(k) + \nabla_{ty} F(y_k; t_k) \right).
\]

Therefore, node \( i \) can compute its prediction direction \( d_{i}(k) \) by having access to the prediction directions \( d_{i}(k-1) \) of itself and its neighbors. By initializing the prediction directions at step \( k \) as \( d_i^{(0)}(k) = [H_k^{-1}](0)^{i1} \nabla_{ty} F(y_k; t_k) = (D_k^{(i)})^{-1} \nabla_{ty} F(y_k; t_k) \), nodes can compute their \( K \) level prediction direction \( d_i^{(0)}(0) \) by \( K \) times recursively computing (15). Notice that according to (2), the local

\(^1\)As part of the convergence analysis, we show that \( \|X\| < 1 \), and therefore the series converges as indicated.
Algorithm 1 Approximate prediction direction for node $i$

Input: Gradient $\nabla_{y_p} F(y_k; t_k)$, matrices $D_k^i$ and $B_k^i$ for $j \in \mathcal{N}_i, j \neq i$

1: Compute the initial prediction direction $d_k^{i(0)} = \left(D_k^i\right)^{-1} \nabla_{y_p} F(y_k; t_k)$ for $\tau = 0, 1, \ldots, K - 1$.
2: Exchange the prediction direction $d_k^{i(\tau)}$ with neighbors $j \in \mathcal{N}_i$.
3: Compute the updated prediction direction $d_k^{i(\tau + 1)}$ as in (15).

Output: Return the approximate prediction direction $d_k^{i(K)}$.

Algorithm 2 DePCoT at node $i$

Require: Initial variable $y_k^0 \in \mathbb{R}^p$ for $k = 0, 1, 2, \ldots$

1: Compute the block: $D_k^i = \nabla_{y_p} \Phi(y_k; t_k) + L I$.
2: Exchange the decision variable $y_k^i$ with neighbors $j \in \mathcal{N}_i$.
3: Compute the blocks $B_k^i$ and $B_k^j$ as in (18) and (19).
4: Compute the mixed derivative $\nabla_{y_p} F(y_k; t_k)$ as in (16).
5: Compute the prediction direction $d_k^{i(K)}$ by Algorithm 1.
6: Execute the prediction step: $y_k^{i+1} = y_k^i - h d_k^{i(K)}$.
7: Exchange the predicted variable $y_k^{i+1}$ with neighbors $j \in \mathcal{N}_i$.
8: Compute the gradient $\nabla_{y_p} F(y_k^{i+1}; t_k^{i+1})$ as in (21).
9: Execute the correction step: $y_k^{i+1} = P_L y_k^{i+1} + \gamma \nabla_{y_p} F(y_k^{i+1}; t_k^{i+1})$.

The correction step (13) is also decentralized given the assumption on $\mathcal{N}$ [cfr. Assumption 1]. In particular, By defining components $\nabla_{y_p} F(y; t)^i \in \mathbb{R}^p$ of the objective function gradient $\nabla_{y_p} F(y; t) = [\nabla_{y_p} F(y; t)^1, \ldots, \nabla_{y_p} F(y; t)^p] \in \mathbb{R}^{p \times N}$, the update in (13) can be decomposed into local components as

$$y_k^{i+1} = P_L y_k^{i+1} + \gamma \nabla_{y_p} F(y_k^{i+1}; t_k^{i+1})$$

(20)

The DePCoT method is summarized in Algorithm 2, while in Algorithm 1 we have summarized the approximate prediction direction computation. As for Algorithm 2, steps 1-4 as well as 7-8 are preliminary communication and computation steps in order to compute the prediction and correction steps. Steps 5-6 contain the approximate prediction step, while step 7 implements the correction step. As for Algorithm 1, steps 0-1 are preliminary computation and communication steps, while step 2 computes the approximate prediction direction per node.

III. CONVERGENCE ANALYSIS

In this section, we study the convergence properties of DePCoT. We show that as time passes the sequence of variable $y_k$ approaches a neighborhood of the optimal argument $y^*(t_k)$. In proving the results we make the following assumptions.

Assumption 1: There exists a set $Y = Y_1 \times \cdots \times Y^n \subseteq \mathbb{R}^p$ whose interior contains the optimal argument trajectory $y^*(t)$ of (1) for each $t$, i.e., $y^*(t) \in \text{int}(Y)$, for $t \geq 0$.

Assumption 2: The local functions $\phi^i$ are twice differentiable and the eigenvalues of the Hessians $\nabla_{y_p} F(y^i; t)$ are bounded by constants $0 < m$ and $M < \infty$. Therefore, the eigenvalues of the aggregate function $\Phi(y; t) := \sum_{i=1}^n \phi^i(y^i; t)$ are bounded uniformly as

$$m I \preceq \nabla_{y_p} \Phi(y; t) \preceq M I$$

(22)

Assumption 3: The functions $g^{i,j}(y^i; y^j; t)$ are twice differentiable and the eigenvalues of the aggregate function Hessian $\nabla_{y_p} G(y; t)$ are bounded by constants $0 < m$ and $L < \infty$.

$$0 \preceq \nabla_{y_p} G(y; t) \preceq L I$$

(23)

Assumption 4: The derivatives of the global cost $F(y; t)$ defined in (1) are bounded for all $y \in \mathcal{Y}$, as follows:

$$\|\nabla_{y_p} F(y; t)\| \leq C_0, \|\nabla_{y_p} G(y; t)\| \leq C_1, \|\nabla_{y_p} F(y; t)\| \leq C_2$$

(24)

Assumption 1 is a weak assumption and for the case that the set $Y$ is $\mathbb{R}^p$, we only assume the existence of a solution for (1) at each time $t$. However, it is very useful in practice, when we know a priori that the solution trajectory has to be, for instance, positive. The bounds on the eigenvalues of Hessians $\nabla_{y_p} \Phi(y; t)$ and $\nabla_{y_p} G(y; t)$ in Assumptions 2 and 3, respectively, follow that the eigenvalues of the global cost Hessian $\nabla_{y_p} F(y; t)$ are uniformly bounded as $m I \preceq \nabla_{y_p} F(y; t) \preceq (L + M) I$. This bounds besides guaranteeing that Problem (1) is strongly convex and has a unique solution for each time instance, implies that the Hessian $\nabla_{y_p} F(y; t)$ is invertible. Conditions imposed on the higher order derivatives of $F$ in Assumption 4 are often required in time-varying optimization to prove convergence [5], [7], [8].

In the following theorem we show linear convergence of the sequence of variables $y_k$ to a neighborhood of $y^*(t_k)$.

Theorem 1: Consider the DePCoT algorithm defined in (9)-(20). Let Assumptions 1-4 hold and define $\rho$ and $\sigma$ as

$$\rho := \max\{|1 - \gamma m|, |1 - \gamma(L + M)|\}, \quad \sigma := 1 + h \left(\frac{C_0 C_1 + C_2}{m^2 + m^2}\right)$$

(25)

If the sampling interval $h$ and the stepsize $\gamma > 0$ are chosen properly to satisfy the condition $\rho < 1$, then the sequence $y_k$ converges Q-linear to $y^*(t_k)$ up to a bounded error as

$$\|y_k - y^*(t_k)\| \leq (\rho \sigma)^k \|y_0 - y^*(t_0)\| + \frac{h \Gamma(\rho) + O(h^2)}{1 - \rho}$$

(26)

where the function $\Gamma$ and the parameter $\rho$ are defined as

$$\Gamma(\rho) = \frac{C_0 \rho^{K+1}}{m(m + L)(1 - \rho)}, \quad \sigma = \frac{L}{m + L}$$

(27)

Theorem 1 states that the sequence of variables $y_k$ generated by DePCoT converges linearly to a neighborhood of $y(t_k)$ where
the error bound is proportional to $\Gamma(q)h + O(h^2)$. Hence, for any level $K$ of Hessian inverse approximation in DePCoT the error bound of order $O(h)$ is achievable. In addition, by choosing large enough approximation level $K$ we can decrease $\Gamma(q)$ in (27) and push towards the order of $h$ to get the error bound of order $O(h^2)$, for a specific interval of sampling period $h$.

In particular, if $K$ is chosen as $K \geq \left\lfloor \log h / \log q - 1 \right\rfloor$, we obtain $O(h^2)$ asymptotical error bound, which is smaller relative to the $O(h)$ error bound of running algorithms. It has to be noticed that, since typically $K$ is fixed a priori, this $O(h^2)$ bound is achieved only in the interval of $h$ for which $K$ satisfies the aforementioned condition. In addition, the higher the $K$ the more is the communication burden. A thorough characterization is left as future research.

### IV. Numerical Evaluation

We consider a wireless sensor network estimating the intensity of a two dimensional spatial circular wave. The location of the source is $\xi^0 = [1.2, 1.2]$, while its space-time intensity at any location and at any time is

$$c(\xi; t) = \cos(2\pi \omega(t - |\xi - \xi^0|/v)/(4\pi |\xi - \xi^0|)),$$

where $\xi$ is the space location in $\mathbb{R}^2$, while $\omega$ and $v$ are the frequency and velocity of the wave, respectively. Each sensor node is located in the position $\xi^i$ and estimates the intensity of the wave as,

$$\hat{c}(\xi^i; t) = c(\xi^i; t) + \eta^i, \quad \eta^i \sim \mathcal{N}(0, q),$$

with $q$ is a given noise covariance. We formulate the estimation as a least-squares problem with spatial regularizer,

$$\min_{\{y^i, \ldots, y^N\}} \sum_{i=1}^N \frac{1}{2q} \left\| y^i - \hat{c}(\xi^i; t) \right\|^2 + \frac{\beta}{q} \sum_{j \in \mathcal{N}_i} \frac{1}{2} \left\| y^i - y^j \right\|^2$$

for which, $\beta > 0$ is a tuning parameter, and the regularization term serves to push closely located sensors to estimate similar intensities. For this purpose the weight $w^{ij}$ is chosen as $w^{ij} = \exp(-\alpha |\xi^i - \xi^j|)/\delta$, where the parameter $\delta$ is the maximum degree of the nodes of the network, and $\alpha = -\log(0.5)/d$, where $d$ is the maximum communication range distance.

In this numerical example, we consider $n = 500$ sensor nodes located in the square $[-1, 1]^2$. We set $d = \sqrt{2}/\sqrt{n}$. We notice that the communication graph does not have to be connected. The other parameters are: $\omega = 0.1, v = 0.05, \sqrt{\pi} = h$, and $\delta = 13$. To see (30) as an instance of (2), it is sufficient to equate

$$f_i(y^i, t) = \frac{1}{2} (1 - \beta w^{ii}) \left\| y^i \right\|^2 - \hat{c}(\xi^i; t) y^i, \quad w^{ii} = \sum_{j \in \mathcal{N}_i} w^{ij}$$

$$g^{ij}(y^i, t) = \beta \frac{w^{ij}}{2} \left\| y^i \right\|^2, \quad g^{ij}(y^i, y^j, t) = \beta \frac{w^{ij}}{2} y^i - y^j \left\| y^i - y^j \right\|^2,$$

and it can be seen that Assumptions 1 till 4 hold. In the simulations, we use $\beta = 0.25$, and with this $m \geq 1 - 2\beta = .5, L \leq 2\beta = .5, C_0 = .1768, C_1 = 0, C_2 = 0, C_3 = 0.1111$. We use the problem (30) to test the performance of different time-varying optimization algorithms. In particular, (i) A running gradient method (meaning Algorithm 2 without the prediction step); (ii) Our Algorithm 2 with backward approximate time derivative and $K$ communication steps for the computation of the approximate Hessian inverse; (iii-iv) The running dual decomposition method and the running ADMM algorithm of [7] and [8], respectively, adapted to our networked scenario, where we perform dual decomposition/ADMM instead of gradient descent in the correction step (and no prediction).

### References

[1] J. Koshal, A. Nedić, and U. Y. Shanbhag, “Multiuser Optimization: Distributed Algorithms and Error Analysis,” SIAM Journal on Optimization, vol. 21, no. 3, pp. 1046 – 1081, 2011.

[2] A. Beck, A. Nedić, A. Ozdaglar, and M. Teboulle, “An $O(1/k)$ Gradient Method for Network Resource Allocation Problems,” IEEE Transactions on Control of Network Systems, vol. 1, no. 1, pp. 64 – 73, 2014.

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

[4] 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.

[5] 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.

[6] S.-Y. Tu and A. H. Sayed, “Mobile Adaptive Networks,” IEEE Journal of Selected Topics in Signal Processing, vol. 5, no. 4, pp. 649 – 664, 2011.

[7] 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.

[8] 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.

[9] A. Simonetto, A. Mokhtari, A. Koppel, G. Leus, and A. Ribeiro, “Prediction-Correction Methods for Networked Time-Varying Convex Optimization.” In preparation, 2015.

[10] A. Simonetto, “A Class of Prediction-Correction Methods for Time-Varying Convex Optimization,” Submitted http://arxiv.org/abs/1509.05196, 2015.

[11] A. Mokhtari, Q. Ling, and A. Ribeiro, “Network Newton-Part I: Algorithm and Convergence,” arXiv preprint arXiv:1504.00677, 2015.

[12] A. Mokhtari, “Network Newton-Part II: Convergence Rate and Implementation,” arXiv preprint arXiv:1504.01620, 2015.