Decentralized optimization over time-varying graphs

Alexander Rogozin\textsuperscript{1}, Alexander Gasnikov\textsuperscript{1,2,3}, Aleksander Beznosikov\textsuperscript{4}, Dmitry Kovalev\textsuperscript{4}

1. Moscow Institute of Physics and Technology, Moscow, Russia
2. Caucasus Mathematical Center, Adyghe State University, Maikop, Russia
3. IITP RAS, Moscow, Russia
4. Université catholique de Louvain (UCL), CORE (IMMAQ), Louvain-la-Neuve, Belgium

Decentralized optimization over time-varying networks has a wide range of applications in distributed learning, signal processing and various distributed control problems. The agents of the distributed system locally hold optimization objectives and can communicate to their immediate neighbors over a network that changes from time to time. In this paper, we survey state-of-the-art results and describe the techniques for optimization over time-varying graphs. We also give an overview of open questions in the field and formulate hypotheses and directions for future work.

1 Introduction

We consider a sum-type optimization problem

\[ \min_{x \in \mathbb{R}^d} f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(x). \]

The nodes can exchange information through a decentralized communication network, which may be time-varying. Each node is connected to several others via communication links and can communicate to them. A time-varying network is represented as a sequence of graphs (this paper focuses on undirected graphs). Since there is no centralized aggregator (server node, master node), each agent locally holds a copy \( x_i \) of the decision vector \( x \). The vectors held by the nodes should be synchronized, but the agents can communicate only to their immediate neighbors.

Problems of type (1) arise in many applications where centralized aggregation is limited due to the structure of the network, privacy constraints or large amounts of data. Applications include vehicle coordination and control, distributed statistical inference and machine learning \cite{50, 15, 41}, power system control \cite{51, 16}, distributed averaging \cite{9, 47, 69}, formation control \cite{45, 52}, distributed spectrum sensing \cite{5}. See surveys \cite{44, 38} for additional examples.

Each \( f_i \) is assumed to be convex and stored at a separate computational agent (or node). Moreover, we assume \( f_i \) is equipped with a first-order oracle (either stochastic or deterministic). That means that each computational node can compute gradients or stochastic gradients of the function it holds. During the computation process, the agents can exchange their decision vectors and gradients.

A decentralized optimization algorithm should be designed in such a way that the sum of functions is minimized while the decision vectors held at different computational nodes stay approximately the same. Assuming that node \( i \) holds \( x_i \), the optimal point in the decentralized sense should be consensual and optimal, i.e.

\[ x_1 = \ldots = x_m = x^* = \arg \min_{x \in \mathbb{R}^d} \frac{1}{m} \sum_{i=1}^{m} f_i(x). \]

1.1 Notation

Let us denote the column vector \( \text{col}(x_1, \ldots, x_m) = (x_1^{\top} \ldots x_m^{\top})^{\top} \).

1.2 Solution accuracy

We rewrite problem (1) as

\[ \min_{x \in \mathbb{R}^d} F(x) = \sum_{i=1}^{m} f_i(x_i), \quad \text{s.t.} \quad x_1 = \ldots = x_m. \]

Note that \( \nabla F(x) = \text{col}(\nabla f_1(x_1) \ldots \nabla f_m(x_m)) \).

We call \( x \) an \( \varepsilon \)-solution if

\[ \frac{1}{m} \sum_{i=1}^{m} f_i(x_i) - f^* \leq \varepsilon, \quad \mathbb{E} \left[ \frac{1}{m} \sum_{i=1}^{m} \| x_i - \hat{x}_i \|^{1/2} \right] \leq C \varepsilon. \]

Here \( C \) is some constant not dependent on \( \varepsilon \). The constant \( C \) may differ for different algorithms. Typically for dual methods constant \( C \) depends on the norm of the dual problem solution. For each primal method, constant \( C \) is individual.

For stochastic methods, the output \( x \) is stochastic, and by \( \varepsilon \)-accuracy we understand

\[ \mathbb{E} \left[ \frac{1}{m} \sum_{i=1}^{m} f_i(x_i) \right] - f^* \leq \varepsilon, \quad \mathbb{E} \left[ \frac{1}{m} \sum_{i=1}^{m} \| x_i - \hat{x}_i \|^{1/2} \right] \leq C \varepsilon. \]

The complexity of distributed algorithms is measured by two quantities: the number of communication rounds and the number of oracle calls performed by a separate node.

1.3 Paper organization

In Section 2 we discuss the consensus over time-varying graphs and formulate the assumptions on the communication network.
Section 2 covers the main techniques and state-of-the-art results in distributed optimization. After that, in Section 3 we describe the algorithms for decentralized saddle-point problems.

2 Time-varying consensus

The nodes communicate through a time-varying communication network. This paper is focused on undirected graphs. The network is represented as a sequence of graphs \( \{G^k\}_{k=0}^{\infty} \). We will also refer to \( \{G^k\}_{k=0}^{\infty} \) as a time-varying graph. The problems of reaching a consensus over time-varying graphs has been studied since 1980’s (see i.e. seminal works \([66, 68]\)).

In order to maintain consensus constraints \( x_1 = \ldots = x_m \) in problem \(4\), a sequence of communication matrices is assigned to the time-varying graph. This paper focuses on undirected graphs, and the two most used types of communication matrices are mixing matrices and gossip matrices. Typically, mixing matrices are used in methods that use primal oracle and gossip matrices are employed in dual algorithms. Both types of matrices are defined in such way that a matrix-vector multiplication corresponds to one synchronized communication round.

2.1 Mixing matrix

**Assumption 1.** Mixing matrix sequence \( \{W^k\}_{k=0}^{\infty} \) satisfies the following properties.

- (Decentralized property) \( W^k \cdot I = I \) and \( I^T W^k = 1^T \).
- (Decentralized property) \( W^k \cdot I = I \) and \( I^T W^k = 1^T \).
- (Spectrum property) There exists a positive \( \tau \in \mathbb{Z} \) and \( \chi > 0 \) such that for any \( k \geq \tau - 1 \) and any \( x \in \mathbb{R}^d \) we have
  \[
  \left\| \left( W^k - \frac{1}{m} 1 1^T \right) x \right\|^2 \leq \left( 1 - \frac{\tau}{\chi} \right) \left\| x \right\|^2,
  \]
  where \( W^0 = \cdots \) ... \( W^{k-\tau} \).

**Remark 2.** Our definition of \( \chi \) is slightly different from the one typically used in the literature. For example, \([40, 25]\) assume

\[
\left\| \left( W^k - \frac{1}{m} 1 1^T \right) x \right\|^2 \leq \left( 1 - \frac{\tau}{\chi} \right) \left\| x \right\|^2.
\]

We need our definition (let us call it reduced condition number of the time-varying network) solely for simplicity of notation: it allows to write \( \chi \) instead of \( \tau \chi \) in all complexity bounds. In the particular case of \( \tau = 1 \) (i.e. when the graph stays connected at every iteration), the definitions coincide.

Also for each \( k \) we introduce \( W^k = W^k \otimes I \). The spectrum property in Assumption \(4\) ensures geometric convergence of consensus iterates \( x^{k+1} = W^k x^k \) to consensus point \( \overline{x} \). After \( N = O\left( \log \left( \frac{1}{\varepsilon} \right) \right) \) iterations we have \( \| x^N - \overline{x} \| \leq \varepsilon \). Here and below we skip the dependence from different parameters except \( \varepsilon \) under log \( \cdot \).

**Remark 3.** Sufficient conditions for Assumption \(4\). Paper \([40]\) gives sufficient conditions for Assumption \(4\) to hold. Firstly, the graph sequence \( \{G^k\}_{k=0}^{\infty} \) should be \( \tau \)-connected. That means that for any \( k \geq 0 \) the union of \( \tau \) consequent graphs \( G^k = \{V, \cup_{i=1}^{\tau} E^i\} \) is connected. Secondly, the following restrictions on the mixing matrix weights are imposed:

1. (Double stochasticity) \( W^k \cdot I = I \) and \( I^T W^k = 1^T \).
2. (Positive diagonal) \( W^k \cdot i \) > 0 for \( i = 1, \ldots, m \).
3. (Edge utilization) If \( (i, j) \in E^k \), then \( W^k [1] > 0 \), else \( W^k [1] = 0 \).
4. (Nonvanishing weights) There exists \( \theta > 0 \) such that if \( W^k [1] > 0 \), then \( W^k [1] \geq \theta \).

In other words, the term \( \tau \) in the Spectrum property of Assumption \(1\) describes the number of iterations such that the union of \( \tau \) consequent graphs is connected.

Typically mixing matrices are used in primal algorithms.

A possible way to build mixing matrices satisfying Assumption \(4\) is to use Metropolis weights \([40]\):

\[
W^k [1] = \begin{cases} \frac{1}{\max(\text{deg}(i), \text{deg}(j), \tau)} & \text{if } (i, j) \in E^k, \\ 0 & \text{if } (i, j) \not\in E^k \text{ and } i \neq j, \\ 1 - \sum_{\neq [1]} W^k [1], & i = j \end{cases}
\]

2.2 Gossip matrix

Dual methods typically use a notation of a gossip matrix sequence \( \{L^k\}_{k=0}^{\infty} \).

**Assumption 4.** Gossip matrix sequence \( \{L^k\}_{k=0}^{\infty} \) satisfies the following properties.

1. \( L^k [1] = 0 \) if \( i \neq j \) and \( (i, j) \not\in E^k \).
2. \( L^k \subseteq \text{span}(1) \).
3. \( \text{Im } L^k \subseteq \{ x \in \mathbb{R}^m : x_1 + \ldots + x_m = 0 \} \).
4. There exists a positive \( \tau \in \mathbb{Z} \) and \( \chi > 0 \) such that for any \( k \geq \tau - 1 \) it holds

\[
\| L^k x - x \|^2 \leq \left( 1 - \frac{\tau}{\chi} \right) \| x \|^2
\]

for all \( x \in \mathbb{R}^m \) such that \( x_1 + \ldots + x_m = 0 \). Here \( L^k \) is defined as

\[
I - L^k = (I - L^k) \ldots (I - L^{k-\tau+1}).
\]

For time-static networks, let \( L \) denote the gossip matrix. Consensus constraints in problem \(3\) can be written as \( L x = 0 \). Therefore, distributed optimization is formulated as an affinely constrained problem. The dual approach build upon optimization of the function dual to \( F \) s.t. \( L x = 0 \).

A possible way to obtain a gossip matrix is \( L^k = I - W^k \).

As noted in \([25]\), mixing matrix sequence \( \{W^k\}_{k=0}^{\infty} \) satisfies Assumption \(4\) if and only if gossip matrix sequence \( \{L^k = I - W^k\}_{k=0}^{\infty} \) satisfies Assumption \(4\).

Alternatively, a gossip matrix can be built using a graph Laplacian. We call \( L(G^2) \) a Laplacian of \( G^2 \) if

\[
L(G^2) = \begin{cases} \text{deg}(i), & i = j, \\ -1, & (i, j) \in E^2, \\ 0, & (i, j) \not\in E^2 \text{ and } i \neq j. \end{cases}
\]
Then \( \mathcal{L}^k = \mathbf{L}(G^k)/\lambda_{\text{max}}(\mathbf{L}^k) \) satisfies Assumption 4. Moreover, in the case \( \tau = 1 \) we can equivalently define \( \chi \) as

\[
\chi = \sup_{k \geq 0} \frac{\lambda_{\text{max}}(\mathbf{L}(G^k))}{\lambda_{\text{min}}(\mathbf{L}(G^k))}.
\]

### 2.3 Multi-step consensus

Let us discuss the multi-step consensus procedure. Consider a problem with \( L \)-smooth and \( \mu \)-strongly convex surrogates \( f_i \) distributed over network satisfying Assumption 1 with parameters \( \tau, \chi \). Let every iteration of the method corresponds to \( O(1) \) local computations and \( O(T) \) communications. To reach \( \varepsilon \)-accuracy one needs to perform \( O\left( \chi(T)/\tau \sqrt{L/\mu} \log(1/\varepsilon) \right) \) local computations and \( O\left( \chi(T) \sqrt{L/\mu} \log(1/\varepsilon) \right) \) communications. However, the lower bound on the number of oracle calls is \( O\left( \sqrt{L/\mu} \log(1/\varepsilon) \right) \). In order to control communication and computation complexities, a multi-step technique is used.

On the one hand, we can choose \( T = T_1 = \tau \). In this case \( \chi(T_1) = \chi \) and we have communication complexity \( O\left( \chi \sqrt{L/\mu} \log(1/\varepsilon) \right) \) and local computation complexity \( O\left( \chi \tau \sqrt{L/\mu} \log(1/\varepsilon) \right) \). In the case of \( \tau = 1 \) the complexities coincide, which corresponds to a single-step consensus, i.e., performing one communication step after each computation.

On the other hand, setting \( T = T_2 = \lfloor \chi \log 2 \rfloor \), we have \( \chi(T_2) = O(1) \). Therefore, the required number of communications is \( O\left( \chi \sqrt{L/\mu} \log(1/\varepsilon) \right) \) and the number of oracle calls is \( O\left( \sqrt{L/\mu} \log(1/\varepsilon) \right) \).

If \( \tau \geq 2 \), both \( T_1 = \tau \) and \( T_2 = \lfloor \chi \log 2 \rfloor \) lead to making several consensus iterations after each oracle call. However, in the case \( \tau = 1 \) we note that \( T = T_1 \) refers to a single-step consensus while \( T = T_2 \) describes to a multi-step consensus. The case \( \tau = 1 \) was historically the first to be studied in the literature since it particularly covers time-static graphs, and we inherit the term multi-step consensus from it. Therefore, we call the scheme with \( T = T_2 \) a multi-step consensus even when \( \tau \geq 2 \).

Multi-step communication procedure is used in distributed optimization over static graphs, as well. In the time-static case such technique is called Chebyshev acceleration and it allows to additionally reduce communication complexity to \( \sqrt{\chi} \).

Given a gossip matrix \( W \) with condition number \( \chi \), let us replace \( W \) by a Chebyshev polynomial \( P_K(W) \) of degree \( K = \lfloor \chi \rfloor \). The condition number of \( P_K(W) \) equals \( O(1) \) due to the specific structure of the polynomial (see [57]). Moreover, loopless Chebyshev acceleration proposed in [59] is achieved without explicit multi-step consensus.

The acceleration on communication steps is not possible in the time-varying scenario, i.e., we cannot reduce \( \chi \) to \( \sqrt{\chi} \) in the time-varying setting. This follows from lower complexity bounds [25].

Reaching the consensus over time-varying networks can be viewed as a quadratic optimization problem with a time-varying objective function \( \ell^k(x) = \frac{1}{2} \| x - W x \|_2^2 \). All functions \( \ell^k(x) \) have the same minimizer \( x^* \). Non-accelerated gradient descent corresponds to a consensus algorithm. On each iteration of gradient descent, we have the contraction of potential function \( \Phi^k = \frac{1}{2} \| x^k - x^* \|_2^2 \). This contraction is robust to graph changes which makes non-accelerated consensus converge over time-varying graphs.

On the contrary, accelerated gradient methods build upon a potential function of type \( \Phi^k = a_k (c(x^k) - c^*) + b_k \| x^k - x^* \|_2^2 \) [2], where \( y^k, x^k \) are additional extrapolation sequences and \( a_k, b_k > 0 \) are scalars (note that in our case optimal value \( c^* = 0 \)). The first summand in \( \Phi^k \) contains the function value and therefore is not robust to network changes.

### 3 Optimization

In this section, we cover several classes of objective functions. We describe the results for smooth and non-smooth (strongly) functions with deterministic and stochastic gradients. Both algorithms that use primal and dual oracle are covered. We also briefly discuss the possible extensions to novel classes of problems: data similarity, directed graphs and alternative assumptions on the time-varying networks.

#### 3.1 Definitions and assumptions

We recall several standard definitions first.

**Definition 5.** Consider function \( h : \mathbb{R}^d \to \mathbb{R} \).

1. Function \( h \) is convex if for any \( x_1, x_2 \in \mathbb{R}^d \) it holds
   \[
h(x_2) \geq h(x_1) + \langle \nabla h(x_1), x_2 - x_1 \rangle.
   \]
2. Function \( h \) is \( \mu \)-strongly convex if for any \( x_1, x_2 \in \mathbb{R}^d \) it holds
   \[
h(x_2) \geq h(x_1) + \langle \nabla h(x_1), x_2 - x_1 \rangle + \frac{\mu}{2} \| x_2 - x_1 \|^2.
   \]
3. Function \( h \) is \( L \)-smooth if for any \( x_1, x_2 \in \mathbb{R}^d \) it holds
   \[
h(x_2) \leq h(x_1) + \langle \nabla h(x_1), x_2 - x_1 \rangle + \frac{L}{2} \| x_2 - x_1 \|^2.
   \]

The assumptions on objective functions are standard for convex optimization.

**Assumption 6.** For every \( i = 1, \ldots, m \), function \( f_i \) is convex.

**Assumption 7.** For every \( i = 1, \ldots, m \), function \( f_i \) is \( \mu \)-strongly convex.

**Assumption 8.** For every \( i = 1, \ldots, m \), function \( f_i \) is \( L \)-smooth.

**Assumption 9.** For every \( i = 1, \ldots, m \), the norm of subgradients of \( f_i \) is bounded by \( M_i \), i.e.,

\[
\| \partial f_i(x) \| \leq M_i.
\]

Further in the paper, we will use Assumptions 6, 7, 8, and 9 in different combinations. We will also cover stochastic case, for which we need the following assumption.

**Assumption 10.** For every \( i = 1, \ldots, m \), function \( f_i \) is equipped with a non-biased stochastic gradient \( \nabla f_i(x, \xi) \) with variance bounded by \( \sigma^2_i \). For any \( x \in \mathbb{R}^d \) it holds

\[
\mathbb{E} \| \nabla f_i(x, \xi) - \nabla f_i(x) \|^2 \leq \sigma^2_i.
\]

Here random variables \( \{\xi_i\}_{i=1}^m \) are independent.
The results in the literature use local, global and original constants characterizing objective functions. By local constants we understand the worst-case constants of \( f_i \). That is,

\[
L_i = \max_{i=1,\ldots,m} L_i, \quad M_i = \max_{i=1,\ldots,m} M_i, \quad \mu_i = \min_{i=1,\ldots,m} \mu_i, \quad \sigma_i = \max_{i=1,\ldots,m} \sigma_i.
\]

(4a)

Note that \( L_i \) and \( \mu_i \) are the smoothness and strong convexity constants of function \( F \) defined in (2), respectively. By global constants we mean the average constants, i.e.

\[
L_g = \frac{1}{m} \sum_{i=1}^{m} L_i, \quad M_g = \frac{1}{m} \sum_{i=1}^{m} M_i, \quad \mu_g = \frac{1}{m} \sum_{i=1}^{m} \mu_i, \quad \sigma_g = \frac{1}{m} \sum_{i=1}^{m} \sigma_i.
\]

(5a)

(5b)

By original constants we understand the constants \( L_i, \mu_i, M_i \) corresponding to function \( f \) itself.

Note that \( L_j \leq L_g \leq L_i, \quad M_j \leq M_g \leq M_i, \quad \mu_j \geq \mu_g \geq \mu_i, \quad \sigma_j \leq \sigma_g \leq \sigma_i \).

Remark 11. Local and global may significantly differ. For example, consider \( f_i(x) = \|x\|^2/2 \) for \( i = 1, \ldots, m - 1 \) and \( f_m(x) = m \|x\|^2/2 \). Then \( L_i = m, \quad L_g = 2 - 1/m \). If the number of nodes is big, we have \( L_j \gg L_g \). Moreover, if \( x \) is restricted to a Euclidean ball \( B_r(0) \) for some \( r > 0 \), we also have \( M_j = m \gg (2 - 1/m) = M_g \).

Note that global and original constants may differ as well. Let \( m = d \), and \( f_i(x) = (x_i^0)^2/2 \), where \( x_i^0 \) is the \( i \)-th component of \( x \). Then \( L_g = 1, \quad L_j = 1/d, \quad L_j/L_j = d \).

Moreover, a trick suggested in [57] can improve the dependence from \( \mu_i \) to \( \mu_j \). Namely, we can replace \( f_i(x) \) by \( \hat{f}_i(x) = f_i(x) + (\mu_i - \mu_j)/2 \|x\|^2 \).

All throughout this section, we let \( R = \|x^0 - x^*\| \) denote the distance from initial guess to solution.

3.2 Algorithm techniques

First, let us show the concepts of decentralized optimization on a simple algorithm. Decentralized gradient descent [72][42][43] is a method that directly combines gradient steps and communication rounds.

\[
x^{k+1} = W^k x^k - \gamma \nabla F(x^k),
\]

(6)

According to decentralized property in Assumption 11 the \( i \)-th node computes \( x^{k+1}_i \) only by communications with its immediate neighbors at time step \( k \). Therefore, \( W^k x^k \) corresponds to one (synchronous) communication round. Schemes of type (4) are relatively simple to analyze (see i.e. analysis via Lyapunov functions in [72]) but do not reach optimal complexity bounds.

We would like to discuss the three techniques that led to different state-of-the-art primal and dual algorithms: inexact oracle, gradient tracking and ADOM.

**Technique 1: gradient tracking.** A group of algorithms uses a gradient tracking technique. This approach assumes that the local gradients at the nodes are averaged as well as decision vectors. For example, let us initialize \( x^0, \quad y^0 = \nabla F(x^0) \) and run procedure [40]

\[
x^{k+1} = W^k x^k - \gamma \nabla F(x^k), \quad y^{k+1} = W^k y^k + \nabla F(x^{k+1}) - \nabla F(x^k).
\]

(7a)

(7b)

Here \( y^k \) stands for gradient approximation. Gradient tracking allows each node to have a local approximation \( y^k \) of the average gradient over the network. The technique was successfully applied to time-varying networks in [40] to show first geometric convergence rates for time-varying graphs. After that, gradient tracking allowed to reach lower complexity bounds by Acc-GT algorithm proposed in [31]. The paper [71] also introduced Mudag that reached optimal bounds up to \( \log(L_i/\mu_i) \) factor. Gradient tracking algorithms have a non-complicated structure but their analysis is quite involved.

It is also possible to use different mixing matrices for vectors and gradients as in push-pull gradient methods [49][45]. Push-pull methods are capable of working over directed graphs.

**Technique 2: ADOM.** We name this novel technique after the algorithm where it was originally used [27]. The technique of ADOM combines saddle-point reformulation and error-feedback. Initially ADOM was used for dual oracle in [27] and then was employed for primal oracle in [24]. ADOM and ADOM+ reach lower complexity bounds. From our point of view, the derivation of both algorithms is natural and logical compared to gradient tracking methods. On the other hand, the algorithms have quite a complicated structure.

**Technique 3: inexact oracle.** One of the possible ways to develop a distributed algorithm is based on inexact oracle concept [12]. After the nodes make a local computation step, the decision vectors held by the nodes are averaged up to target accuracy \( \epsilon \) by running multi-step consensus for \( T = O(\log(1/\epsilon)) \) iterations. According to Assumption 11 this allows to get a projection on the consensus constraint set \( x_1 = \ldots = x_n \) with accuracy \( \epsilon \). As a result, at each computation step the values at the nodes are approximately averaged, and therefore the gradient is obtained with approximation accuracy \( \epsilon \). The inexact oracle concept allows to tackle the inexactness of the gradient.

Initially the inexact oracle technique was developed for time-static networks in [19]. Its generalization on time-varying networks for accelerated gradient method were made in [55][44][40].

The main advantage of using the inexact oracle framework is the simplicity of interpretation and analysis of the distributed method. However, all such schemes have an additional logarithmic term in communication complexity (i.e. the complexity for strongly convex smooth objectives is proportional to \( \log^2(1/\epsilon) \)). Inexact oracle schemes need to perform a comparatively large number of communications between steps related to other techniques, and the number of communications needs to be accurately tuned.

**Regularization.** Some of the papers cover only the strongly-convex case. Such results can be generalized to a non-strongly convex problems using regularization.

**Lemma 12 (Regularization).** [77] Let \( h(x) \) be a convex function and \( \epsilon > 0 \) be the desired accuracy. Let \( h(x) \) have a minimizer \( x^* \) such that \( \|x^*\| \leq R \). Consider regularized function
\( \overline{h}(x) = h(x) + \frac{\mu}{2} \| x \|^2 \). Let \( \tilde{x} \) be an \( \epsilon/2 \)-minimizer of \( \overline{h} \), i.e.

\[
\overline{h}(\tilde{x}) - \overline{h}^* \leq \frac{\epsilon}{2}.
\]

Then we have \( h(\tilde{x}) - h^* \leq \epsilon \).

In the following tables, we cover the existing results as well as their possible extensions. Some extensions can be made by regularization, and some are just guesses based on distributed optimization over time-static graphs. Our hypotheses are marked in blue.

### 3.3 Primal oracle

We begin with primal algorithms and cover deterministic and stochastic cases. Lower bounds for problems on time-varying graphs satisfying Assumptions \([12]\) and \([8]\) are

\[
\Omega \left( \sqrt{\frac{L_h}{\mu_h}} \log \frac{1}{\epsilon} \right) \text{ communications,}
\]

\[
\Omega \left( \sqrt{\frac{L_\ell}{\mu_\ell}} \log \frac{1}{\epsilon} \right) \text{ local oracle calls}
\]

as shown in \([25]\).

The constant \( C \) used in \([3]\) to measure consensus is individual for every primal method. For example, for stochastic APM-C \([54]\) we have \( C = (m/2) \mu_h / (L_h^2 L_\ell^2) \).

The primal approach assumes that every node \( i \) has access only to the gradient of (primal) function \( f_i \).

The following table describes the results for primal oracle as well as our hypotheses (in blue). We omit \( O(\cdot) \) notation for brevity.

|       | convex | str. convex |
|-------|--------|-------------|
| smooth | Acc-GT \([11]\) | Acc-GT \([11]\) |
|       | \( \chi \sqrt{\frac{L_{\ell}^2}{\mu_{\ell}}} \log \frac{1}{\epsilon} \) | \( \chi \sqrt{\frac{L_{\ell}^2}{\mu_{\ell}}} \log \frac{1}{\epsilon} \) |
|       | \( \sqrt{\frac{L_{\ell}^2}{\mu_{\ell}}} \log \frac{1}{\epsilon} \) | \( \sqrt{\frac{L_{\ell}^2}{\mu_{\ell}}} \log \frac{1}{\epsilon} \) |
| non-smooth | \( \chi \frac{M_{i,R}}{\mu_{i,R}} \log \frac{1}{\epsilon} \) | \( \chi \frac{M_{i,R}}{\mu_{i,R}} \log \frac{1}{\epsilon} \) |
|       | \( \frac{M_{i,R}^2}{\mu_{i,R}^2} \) | \( \frac{M_{i,R}^2}{\mu_{i,R}^2} \) |

Table 1: Primal oracle

**Deterministic smooth objectives.** The bounds for smooth strongly convex primal scenario are achieved by ADOM+ \([25]\) and Acc-GT \([31]\). These algorithms appeared approximately at the same time and are based on different techniques. Acc-GT uses gradient tracking and Nesterov acceleration, while ADOM+ is based on the independent technique with error feedback and specific problem reformulation. The bounds for a smooth (non-strongly) convex case are obtained via regularizaion according to Lemma \([12]\) which is maid in a straightforward way.

**Deterministic non-smooth objectives.** One of the first papers on distributed subgradient methods \([22]\) proposed a method of type \([6]\) that converges to a \( O(\gamma) \) neighborhood of the solution at a rate of \( O(1/\epsilon) \), where \( \gamma \) denotes the stepsizes.

These results were generalized to directed graphs in \([39]\). For the time-static scenario, optimal algorithms for non-smooth objectives \([13]\) are derived using gradient sliding technique \([29]\). Our guesses are based on the results for time-static setup. The direct application of gradient sliding to time-varying networks is an open question. An extension to functions satisfying Polyak–Lojasiewicz condition was proposed in \([28]\).

Optimization of stochastic objectives over time-varying networks is not studied as good as with deterministic objectives.

**Stochastic smooth objectives.** SGD with gradient tracking was studied in \([48]\); however, it only converges to a neighborhood of the solution at suboptimal rates. Paper \([54]\) proposed a penalty method (APM-C) that is based on inexact oracle call and a communication round \([33, 57, 27]\).

The dual approach is an alternative to primal. It uses a different oracle, i.e. the gradient of the conjugate function.

**Definition 13.** Consider function \( h : \mathbb{R}^d \to \mathbb{R} \). We call \( h^*(y) \) a conjugate of \( h \) if

\[
h^*(y) = \max_{x \in \mathbb{R}^d} \{ \langle x, y \rangle - h(x) \}.
\]

The dual approach in decentralized optimization utilizes a gossip matrix to interpret problem \([\ref{prob}]\) as a problem with affine constraints. The constraints are used to build a dual problem and solve it with gradient methods. Let us consider a time-static case with gossip matrix \( \mathcal{L} \). Dual problem to \([\ref{prob}]\) writes as

\[
\min \Phi(y) = \max_{\mathbf{y} \in \mathbb{R}^m} \{ \langle \mathbf{y}, \mathcal{L} \mathbf{x} \rangle - F(\mathbf{x}) \}.
\]

One may see that computation of \( \nabla \Phi(y) \) corresponds to an oracle call and a communication round \([33, 57, 27]\).

For the dual approach over time-static networks, it can be shown that constant \( C \) from \([3]\) used to bound consensus accuracy of the solution has form \( C = 1/M_\mu \).

Typically, the following lemma is used to connect optimization properties of the dual to the properties of the primal problem and therefore obtain convergence rates.
Lemma 14. \cite{73} Let function $h^*$ be a conjugate of $h$. If $h$ is $L$-smooth, then $h^*$ is $(1/L)$-strongly convex. If $h$ is $\mu$-strongly convex, then $h^*$ is $(1/\mu)$-smooth.

For stochastic objectives, we assume that functions $f_i^*$ are equipped with a non-biased stochastic first-order oracle with bounded variance.

Assumption 15. For every $i = 1, \ldots, m$, function $f_i^*$ is equipped with a non-biased stochastic gradient $\nabla f_i^*(y, \xi_i)$ with variance bounded by $\sigma_i^2$. For any $y \in \mathbb{R}^d$ it holds
\[
\mathbb{E}|\nabla f_i^*(y, \xi_i) - \nabla f_i^*(y)|^2 \leq \sigma_i^2.
\]

Here random variables $\xi_i$ are independent.

Analogously we introduce worst-case variance $\sigma_i^2 = \max_{i=1}^m \sigma_i^2$, and average variance $\sigma_i^2 = (1/m) \sum_{i=1}^m \sigma_i^2$.

**Deterministic smooth objectives.** One of the first proposed linearly convergent dual methods for time-varying graphs is PANDA \cite{33} that achieves a suboptimal communication complexity $O\left(\chi^2 (L_\epsilon/\mu)^{3/2} \log(1/\epsilon)\right)$ (using multi-step consensus will replace $\chi^2$ with $\chi$). PANDA applies gradient tracking technique to gradients of the dual function. An optimal dual algorithm is ADOM \cite{27}. ADOM uses a specific reformulation of the problem. In particular, it treats multiplication by gossip matrix as a compression operator and employs an error feedback trick.

**Deterministic non-smooth objectives.** Non-accelerated methods for non-smooth problems with dual oracle were proposed in \cite{63}. After that, ADOM was generalized to strongly convex non-smooth problems \cite{73} by using Moreau–Yosida regularization. This generalization also covers constrained sets and supports non-Euclidean setup for the Wasserstein barycenter problem.

| smooth | convex | str. convex |
|--------|--------|-------------|
| comm.  | $\chi \sqrt{\frac{L_{2,\epsilon}}{\mu}} \log \frac{1}{\epsilon}$ | $\chi \sqrt{\frac{L_{2,\epsilon}}{\mu}} \log \frac{1}{\epsilon}$ |
| comp.  | $\sqrt{\frac{L_{2,\epsilon}}{\mu}} \log \frac{1}{\epsilon}$ | $\sqrt{\frac{L_{2,\epsilon}}{\mu}} \log \frac{1}{\epsilon}$ |

| non-smooth | convex | str. convex |
|------------|--------|-------------|
| comm.      | $\frac{M_\epsilon}{\sqrt{\epsilon}} \log \frac{1}{\epsilon}$ | $\frac{M_\epsilon}{\sqrt{\epsilon}} \log \frac{1}{\epsilon}$ |
| comp.      | $\frac{M_\epsilon}{\sqrt{\epsilon}} \log \frac{1}{\epsilon}$ | $\frac{M_\epsilon}{\sqrt{\epsilon}} \log \frac{1}{\epsilon}$ |

Table 3: Dual oracle

Both ADOM and its generalization can be adapted for non-strongly-convex setup by using regularization (see hypotheses in Table 3). We note that the analysis in \cite{73} suggests that complexity bounds include $M_\epsilon$, but we guess that this dependence may be improved to $M_\epsilon$.

**Stochastic smooth and non-smooth objectives.** Even in the time-static scenario, we are only aware of results for quadratic objectives \cite{75}.

| convex-concave | str. convex-concave |
|----------------|---------------------|
| comm.          | $\chi \sqrt{\frac{L_{2,\epsilon}}{\mu}} \log \frac{1}{\epsilon}$ | $\chi \sqrt{\frac{L_{2,\epsilon}}{\mu}} \log \frac{1}{\epsilon}$ |
| comp.          | $\sqrt{\frac{L_{2,\epsilon}}{\mu}} \log \frac{1}{\epsilon}$ | $\sqrt{\frac{L_{2,\epsilon}}{\mu}} \log \frac{1}{\epsilon}$ |

Table 5: Finite-sum smooth optimization problems

Based on results of \cite{75}, we formulate our hypotheses in Table 4. Note that the bounds \cite{75} use worst-case variance $\sigma_i^2$, but we are optimistic to guess that it is possible to get a bound using average variance $\sigma_i^2$.

Dual stochastic algorithms are an open direction for research.

### 3.5 Finite-sum problems
Consider a finite-sum form of problems (1). Each $f_i(x)$ is now represented as a sum of $n$ functions.

$$\min_{x \in \mathbb{R}^d} \sum_{i=1}^m f_i(x) = \frac{1}{n} \sum_{i=1}^m \frac{1}{n} \sum_{j=1}^n f_j(x). \quad (8)$$

Problems of type (8) can be efficiently solved by stochastic methods that use additional knowledge of the sum-type structure. This class of algorithms is referred to as variance reduction methods \cite{21,1}. These methods compute the full gradient sum over all $f_i$ once in several iterations and make stochastic updates to the full gradient approximation afterwards.

The complexity of variance reduction methods typically depends on the worst-case smoothness constant over summants. Let $L_{ij}$ be the smoothness constant of $f_i$ and introduce

$$L_s = \max_{i=1}^m L_{ij}, \quad (9)$$

In distributed optimization, variance reduction is done for time-static graphs \cite{18}. The generalization to time-varying graphs is an open question. We think that a method for decentralized finite-sum optimization of smooth functions over time-varying networks may be done by combining Katyusha \cite{1} and ADOM/ADOM+ \cite{25} or Acc-GT \cite{31}.

$$\mathbb{E}\|W_k^s x - \bar{x}\|^2 \leq \left(1 - \frac{\tau}{\chi}\right)\|x - \bar{x}\|^2, \quad (10)$$

3.6 Time-varying network as a stochastic sequence
Most of the results of this paper are derived in the deterministic scenario: that is, a mixing (or gossip) matrix sequence $\{W_k\}_{k=0}^\infty$ is deterministic and satisfies Assumption 1. Alternatively, each of the mixing matrices $W_k$ may be seen as a random sample from some distribution $\mathcal{W}$. As suggested in \cite{22}, we can assume that for all $x \in \mathbb{R}^{md}$ it holds
where the expectation is taken over the distributions of $W'$ and indices $t = k, \ldots, k - \tau + 1$. Such assumption broadens the class of networks compared to Assumption 1. In particular, it admits a scheme when only two randomly chosen agents communicate at once (see [3] for details) and also includes Local SGD schemes [60, 67, 32]. Distributed SGD under Assumption 10 was analyzed in [22] for optimization and in [5] for saddle-point problems. Also [63] applied Catalyst to the results of [22] to get acceleration.

### 3.7 Statistical similarity

The method of using convex surrogate functions at local computation was proposed in [62]. Combined with gradient tracking technique, it yields a SONATA algorithm [62, 58] that supports non-convex objectives and works over time-varying directed graphs. Several variants of SONATA [61, 58, 62] also support constrained optimization.

In the undirected time-static case with convex objectives, SONATA exploits statistical similarity of local functions [61, 64]. That is, for any $x$ and $i = 1, \ldots, m$ we have

$$\left\| \nabla^2 f_i(x) - \nabla^2 f(x) \right\| \leq \beta$$

for some $\beta > 0$. Under similarity assumption, SONATA communication rates (in time-static setup) are improved to $O(\sqrt{\beta/\mu} \log(1/\varepsilon))$ [61]. Additionally, with Catalyst acceleration, the rates are further improved to $O(\sqrt{\beta/\mu} \log^2(1/\varepsilon))$ [64]. Optimal algorithms were developed in [23]. We note that exploiting statistical similarity in the time-varying case is an open question. Using SONATA and the technique of convex surrogate functions may be a good starting point in this direction.

### 3.8 Problems with affine constraints

Affinely constrained problems over time-varying networks are still another direction of research. For problems of type

$$\min_{x \in \mathbb{R}^d} \quad f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(x),$$

\text{s.t.} \quad Ax = b$$

in the time-static case a range of algorithms was proposed in [37, 35, 36, 70, 56]. The algorithms are based either on dual decomposition or on primal approaches to affinely constrained minimization [26]. Generalization of the results to time-varying architectures is an open branch of research.

### 4 Saddle-point problems

In this section, we consider sum-type min-max problems.

$$\min_{x \in \mathbb{R}^n} \max_{y \in \mathbb{R}^d} \quad f(x, y) = \frac{1}{m} \sum_{i=1}^{m} f_i(x, y).$$

#### 4.1 Definitions and assumptions

Let us recall the standard definitions for saddle-point problems.

**Definition 16.** Consider function $h(x, y)$. Introduce $z = \text{col}(x, y)$ and a vector field $g(z) = \nabla_h h(x, y)$ associated with function $h(x, y)$.

1. Function $h(x, y)$ is convex-concave if the corresponding vector field $g$ is monotone, i.e. for all $z_1, z_2 \in \mathbb{R}^{d_1 + d_2}$, it holds

   $$\langle g(z_2) - g(z_1), z_2 - z_1 \rangle \geq 0.$$

2. Function $h(x, y)$ is $\mu$-strongly convex-concave if the corresponding vector field $g$ is $\mu$-strongly monotone, i.e. for all $z_1, z_2 \in \mathbb{R}^{d_1 + d_2}$, it holds

   $$\langle g(z_2) - g(z_1), z_2 - z_1 \rangle \geq \mu \|z_2 - z_1\|^2.$$

3. Function $h(x, y)$ is $L$-smooth if the corresponding vector field $g$ is $L$-Lipschitz, i.e. for all $z_1, z_2 \in \mathbb{R}^{d_1 + d_2}$, it holds

   $$\|g(z_2) - g(z_1)\| \leq L \|z_2 - z_1\|.$$

**Assumption 17.** For each $i = 1, \ldots, m$ function $f_i(x, y)$ is convex-concave.

**Assumption 18.** For each $i = 1, \ldots, m$ function $f_i(x, y)$ is $\mu_i$-strongly convex-concave.

**Assumption 19.** For each $i = 1, \ldots, m$ function $f_i(x, y)$ is $L_i$-smooth.

**Assumption 20.** For each $i = 1, \ldots, m$ function $f_i(x, y)$ has a bounded gradient, i.e. there exists $M_i > 0$ such that

$$\|\nabla_y f_i(x, y)\| \leq M_i, \quad \|\nabla_y f(x, y)\| \leq M_i.$$

Similarly to Section 3 we discuss stochastic min-max problems and we need the following assumption.

**Assumption 21.** For each $i = 1, \ldots, m$ function $f_i(x, y)$ is equipped with a non-biased gradient oracle with bounded variance, i.e. there exists $\sigma^2 > 0$ such that

$$\mathbb{E} \left[ \|\nabla_y f_i(x, y, \xi) - \nabla_y f(x, y)\|^2 \right] \leq \sigma^2.$$

Random variables $\xi^i$ are independent.

Similarly to Section 3 we introduce worst-case constants $L_i, M_i, \mu_i, \sigma_\xi$ as in [4], average constants $L, M, \mu, \sigma_\xi$ as in [5] and constants that describe $f$ itself $L_f, M_f, \mu_f$.

For saddle-point problems, we introduce a measure of solution accuracy.

**Definition 22.** Consider function $h(x, y)$, vector field $g(z)$ associated with it and set $Z \subseteq \mathbb{R}^{d_1 + d_2}$. For a point $\hat{z} = \text{col}(\hat{x}, \hat{y}) \in \mathbb{R}^{d_1 + d_2}$, we introduce dual gap as follows.

$$\text{Gap}_Z^\beta(\hat{z}) = \max_{z \in Z} \langle g(z), z - \hat{z} \rangle.$$

In order to measure solution accuracy, we fix $D > 0$. We call $(\hat{x}, \hat{y})$ an $\varepsilon$-solution of a distributed saddle-point problem if

$$\text{Gap}_{D \varepsilon = 0}^\beta(\hat{x}, \hat{y}) \leq \varepsilon, \quad \left(\frac{1}{m} \sum_{i=1}^{m} \left\| \hat{x}_i - x_i \right\|^2 + \left\| \hat{y}_i - y_i \right\|^2 \right)^{1/2} \leq C \varepsilon.$$

Similarly to optimization methods, constant $C$ does not depend on $\varepsilon$, but may be individual for every algorithm. If the output of the method is stochastic, we understand $\varepsilon$-solution as

$$\mathbb{E} \left[ \text{Gap}_{D \varepsilon = 0}^\beta(\hat{x}, \hat{y}) \right] \leq \varepsilon, \quad \mathbb{E} \left[ \left(\frac{1}{m} \sum_{i=1}^{m} \left\| \hat{x}_i - x_i \right\|^2 + \left\| \hat{y}_i - y_i \right\|^2 \right)^{1/2} \right] \leq C \varepsilon.$$
4.2 Results for deterministic and stochastic oracle

**Smooth objectives.** Saddle-point problems with deterministic oracle are studied in a more general case (finite-sum structure with variance reduction) in [24]. The reduction of Algorithm 2 of [24] for deterministic case achieves optimal complexity bounds for deterministic setup.

Also inexact oracle technique was used in [7] to build algorithms optimal up to log(1/ε). On the other hand, adding regularization (Lemma 12) to the results of [24] allows to avoid squared logarithmic factors and gives an algorithm with O(χ(L^2D^2/ε) log(1/ε)) communication and O((L^2D/ε) log(1/ε)) oracle complexities. In Table 6, we give the results for DESM [7] but keep in mind that regularization of [24] is also applicable.

| Table 7: Stochastic saddle-point problems |
|------------------------------------------|
| **convex-concave** | **str. convex-concave** |
| **Alg.2 [24]** | **Alg.2 [24]** |
| **comm.** | **comp.** |
| O(χ(L^2D^2/ε) log(1/ε)) | O(χ(L^2D/ε) log(1/ε)) |
| O((L^2D/ε) log(1/ε)) | O((L^2D^2/ε) log(1/ε)) |

Table 6: Deterministic saddle-point problems

In the stochastic smooth setup, [24] proposed a Decentralized extra-step method (DESM) based on inexact oracle framework. Combined with a mini-batching technique, DESM achieves optimal lower bounds in the stochastic case up to a logarithmic factor log(1/ε).

**Non-smooth objectives.** Non-smooth min-max problems can be solved with a subgradient method [43]. However, it is not optimal in communications and therefore nonsmooth distributed saddles are an open direction of research. Our proposal is trying to generalize gradient sliding for optimization [13] and apply it to saddle-point problems as it was done for time-static graphs in [28]. Our guesses for deterministic and stochastic objectives are presented in Tables 6 and 7.

4.3 Finite-sum problems

Let us consider saddle-point problems with finite-sum structure of every function held at the node

\[
\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{m} \sum_{i=1}^{m} \left( \frac{1}{n} \sum_{j=1}^{n} f_{ij}(x, y) \right).
\]

Analogically to Section 3.5 let f_{ij} be L_{ij}-smooth and define worst-case smoothness constant over summands L_{s} as in [24].

An optimal variance reduction algorithm for saddle-point problems was proposed in [24].

| Table 8: Finite-sum smooth min-max problems |
|---------------------------------------------|
| **convex-concave** | **str.convex-concave** |
| **Alg.2 [24]** | **Alg.2 [24]** |
| **comm.** | **comp.** |
| O(χ^2D^2/ε log(1/ε)) | O(χD/ε log(1/ε)) |
| O(χD^2/ε log(1/ε)) | O(χ^2D/ε log(1/ε)) |

Table 8: Finite-sum smooth min-max problems

We note that it may be possible to replace L_{s} with L_{f} by using importance sampling [11].

4.4 Different strong convexity and strong concavity constants

In Tables 6 and 7 we described the results for min-max problems with same constants of strong convexity and strong concavity. However, strong convexity constant μ_{s} may be different from strong concavity constant μ_{c}. In the non-distributed case, lower bounds for saddle-point problems with different constants of strong convexity and strong concavity were derived in [24]. Optimal algorithms achieving these bounds were developed for saddles with bilinear coupling, i.e. problems of form

\[
\min_{x \in \mathbb{R}^n, y \in \mathbb{R}^n} g(x) + y^T A x - q(y),
\]

where g(x) and q(y) are strongly convex functions with constants μ_{s} and μ_{c}, respectively. For saddles with bilinear coupling, optimal algorithms were developed in [20, 26]. Paper [10] proposed a Catalyst type algorithm for general (non-strongly) convex strongly-concave saddles.

Distributed algorithms for saddle point problems with different strong convexity and strong concavity constants are an open venue for research. Paper [34] uses an inexact oracle framework and a mini-batching technique and reaches lower bounds up to logarithmic factor.

5 Acknowledgments

The authors are grateful to Angelia Nedić, Gesualdo Scutari and Anton Proskurnikov.

The work of A. Rogozin and A. Beznosikov in sections 1, 4 was supported by the strategic academic leadership program «Priority 2030» (Agreement 075-02-2021-1316 30.09.2021).

The work of A. Rogozin and A. Gasnikov in sections 2, 3 was supported by the Ministry of Science and Higher Education of the Russian Federation (Goszadaniye) 075-00337-20-03, project No. 0714-2020-0005.

References

[1] Z. Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 1200–1205, New York, NY, USA, 2017. ACM. arXiv:1603.05933.
[2] N. Bansal and A. Gupta. Potential-function proofs for gradient methods. Theory of Computing, 15(1):1–32, 2019.

[3] J. A. Bazerque and G. B. Giannakis. Distributed spectrum sensing for cognitive radio networks by exploiting sparsity. IEEE Transactions on Signal Processing, 58(3):1847–1862, 2009.

[4] D. Bertsekas and J. N. Tsitsiklis. Parallel and distributed computation: numerical methods, volume 23. Prentice hall Englewood Cliffs, NJ, 1989.

[5] A. Beznosikov, P. Dvurechensky, A. Koloskova, V. S. Ust, and A. Gasnikov. Decentralized local stochastic extra-gradient for variational inequalities. arXiv preprint arXiv:2106.08315, 2021.

[6] A. Beznosikov, A. Rogozin, D. Kovalev, and A. Gasnikov. Near-optimal decentralized algorithms for saddle point problems over time-varying networks. In International Conference on Optimization and Applications, pages 246–257. Springer, 2021.

[7] A. Beznosikov, V. Samokhin, and A. Gasnikov. Distributed saddle-point problems: Lower bounds, optimal algorithms and federated gans. arXiv preprint arXiv:2010.13112, 2021.

[8] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. IEEE transactions on information theory, 52(6):2508–2530, 2006.

[9] K. Cai and H. Ishii. Average consensus on arbitrary strongly connected digraphs with time-varying topologies. IEEE Transactions on Automatic Control, 59(4):1066–1071, 2014.

[10] Y. Carmon, A. Jambulapati, Y. Jin, and A. Sidford. Recapp: Crafting a more efficient catalyst for convex optimization. In International Conference on Machine Learning, pages 2658–2685. PMLR, 2022.

[11] D. Csiha and P. Richtarik. Importance sampling for mini-batches. arXiv preprint arXiv:2105.01587, 2021.

[12] D. Dvinskikh. Decentralized algorithms for wasserstein barycenters. arXiv preprint arXiv:2105.01587, 2021.

[13] D. Dvinskikh and A. Gasnikov. Decentralized and parallel primal and dual accelerated methods for stochastic convex programming problems. Journal of Inverse and Ill-posed Problems, 29(3):385–405, 2021.

[14] P. A. Forero, A. Cano, and G. B. Giannakis. Consensus-based distributed support vector machines. Journal of Machine Learning Research, 11(5), 2010.

[15] L. Gan, U. Topcu, and S. H. Low. Optimal decentralized protocol for electric vehicle charging. IEEE Transactions on Power Systems, 28(2):940–951, 2012.

[16] E. Gorbunov, D. Dvinskikh, and A. Gasnikov. Optimal decentralized distributed algorithms for stochastic convex optimization. arXiv preprint arXiv:1911.07363, 2019.

[17] H. Hendriks, F. Bach, and L. Massoulie. An optimal algorithm for decentralized finite sum optimization. arXiv preprint arXiv:2005.10673, 2020.

[18] D. Jakovetić, J. Xavier, and J. M. Moura. Fast distributed gradient methods. IEEE Transactions on Automatic Control, 59(5):1131–1146, 2014.

[19] Y. Jin, A. Sidford, and K. Tian. Sharper rates for separable min-max and finite sum optimization via primal-dual extragradient methods. arXiv preprint arXiv:2202.04640, 2022.

[20] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In Advances in neural information processing systems, pages 315–323, 2013.

[21] A. Koloskova, N. Loizou, S. Boreni, M. Jaggi, and S. U. Stich. A unified theory of decentralized sgd with changing topology and local updates. ICML 2020, arXiv preprint arXiv:2003.10422, 2020.

[22] D. Kovalev, A. Beznosikov, E. Borodich, A. Gasnikov, and G. Scutari. Optimal gradient sliding and its application to distributed optimization under similarity. arXiv preprint arXiv:2205.15136, 2022.

[23] D. Kovalev, A. Beznosikov, A. Sadiev, M. Persiantsov, P. Richtarik, and A. Gasnikov. Optimal algorithms for decentralized stochastic variational inequalities. arXiv preprint arXiv:2202.02771, 2022.

[24] D. Kovalev, E. Gasanov, A. Gasnikov, and P. Richtarik. Lower bounds and optimal algorithms for smooth and strongly convex decentralized optimization over time-varying networks. Advances in Neural Information Processing Systems, 34, 2021.

[25] D. Kovalev, A. Gasnikov, and P. Richtarik. Accelerated primal-dual gradient method for smooth and convex-concave saddle-point problems with bilinear coupling. arXiv preprint arXiv:2112.15199, 2021.

[26] D. Kovalev, E. Shulgin, P. Richtarik, A. Rogozin, and A. Gasnikov. Adom: Accelerated decentralized optimization method for time-varying networks. arXiv preprint arXiv:2102.09234, 2021.

[27] I. Kuruzov, M. Alkousa, F. Stonyakin, and A. Gasnikov. Gradient-type methods for decentralized optimization problems with polyak-lojasiewicz condition over time-varying networks. arXiv preprint arXiv:1911.08527, 2019.

[28] G. Lan. Gradient sliding for composite optimization. Mathematical Programming, 159(1):201–235, Sep 2016.

[29] H. Li and Z. Lin. Accelerated gradient tracking over time-varying graphs for decentralized optimization. arXiv preprint arXiv:2104.02596, 2021.

[30] T. Lin, S. U. Stich, K. K. Patel, and M. Jaggi. Don’t use large mini-batches, use local sgd. arXiv preprint arXiv:1808.07217, 2018.

[31] M. Maros and J. Jaldén. Panda: A dual linearly converging method for distributed optimization over time-varying undirected graphs. In 2018 IEEE Conference on Decision and Control (CDC), pages 6520–6525. IEEE, 2018.

[32] D. Metelev, A. Rogozin, A. Gasnikov, and D. Kovalev. Decentralized saddle-point problems with different constants of strong convexity and strong concavity. arXiv preprint arXiv:2206.00990, 2022.

[33] I. Necsoara and V. Nedic. Distributed dual gradient methods and error bound conditions. arXiv preprint arXiv:1401.4398, 2014.

[34] I. Necsoara and V. Nedic. On linear convergence of a distributed dual gradient algorithm for linearly constrained separable convex problems. Automatica, 55:209–216, 2015.

[35] I. Necsoara, V. Nedic, and I. Dumitrasche. Parallel and distributed optimization methods for estimation and control in networks. Journal of Process Control, 21(5):756–766, 2011.

[36] A. Nedic. Distributed gradient methods for convex machine learning problems in networks: Distributed optimization. IEEE Signal Processing Magazine, 37(3):92–101, 2020.

[37] A. Nedic and A. Olshevsky. Distributed optimization over time-varying directed graphs. IEEE Transactions on Automatic Control, 60(3):601–615, 2014.

[38] A. Nedic, A. Olshevsky, and V. Shi. Achieving geometric convergence for distributed optimization over time-varying graphs. SIAM Journal on Optimization, 27(4):2597–2633, 2017.

[39] A. Nedic, A. Olshevsky, and C. A. Uribe. Fast convergence rates for distributed non-bayesian learning. IEEE Transactions on Automatic Control, 62(11):5538–5553, 2017.

[40] A. Nedić and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. IEEE Transactions on Automatic Control, 54(1):48–61, 2009.

[41] A. Nedic and A. Ozdaglar. Subgradient methods for saddle-point problems. Journal of optimization theory and applications, 142(1):205–228, 2009.

[42] A. Nedic and A. Ozdaglar. Cooperative distributed multi-agent optimization. Convex optimization in signal processing and communications, 340, 2010.

[43] A. Nedic, D. T. A. Nguyen, and D. T. Nguyen. Ah/push-pull method for distributed optimization in time-varying directed networks. arXiv preprint arXiv:2209.06974, 2022.

[44] A. Olshevsky. Efficient information aggregation strategies for distributed control and signal processing. arXiv preprint arXiv:1009.6035, 2010.

[45] A. Olshevsky. Linear time average consensus on fixed graphs and implications for decentralized optimization and multi-agent control. arXiv preprint arXiv:1411.4186, 2014.
S. Pu and A. Nedić. Distributed stochastic gradient tracking methods. Mathematical Programming, 187(1):409–457, 2021.

S. Pu, W. Shi, J. Xu, and A. Nedić. Push-pull gradient methods for distributed optimization in networks. IEEE Transactions on Automatic Control, 66(1):1–16, 2020.

M. Rabbat and R. Nowak. Distributed optimization in sensor networks. In Proceedings of the 3rd international symposium on Information processing in sensor networks, pages 20–27, 2004.

S. S. Ram, V. V. Veeravalli, and A. Nedic. Distributed non-autonomous power control through distributed convex optimization. In IEEE INFOCOM 2009, pages 3001–3005. IEEE, 2009.

W. Ren. Consensus-based formation control strategies for multi-vehicle systems. In 2006 American Control Conference, pages 6–pp. IEEE, 2006.

R. T. Rockafellar. Convex analysis. Princeton university press, 2015.

A. Rogozin, V. Lukoshkin, A. Gasnikov, and V. Lukoshkin. An accelerated method for decentralized distributed stochastic optimization over time-varying graphs. Conference on decision and control, 2021.

A. Rogozin, V. Lukoshkin, A. Gasnikov, D. Kovalev, and E. Shulgin. Towards accelerated rates for distributed optimization over time-varying networks. In International Conference on Optimization and Applications, pages 258–272. Springer, 2021.

A. Rogozin, D. Yarmoshik, K. Kopylova, and A. Gasnikov. Decentralized strongly-convex optimization with affine constraints: Primal and dual approaches. arXiv preprint arXiv:2207.04555, 2022.

K. Scaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massoulié. Optimal algorithms for smooth and strongly convex distributed optimization in networks. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pages 3027–3036. JMLR. org, 2017.

G. Scutari and Y. Sun. Distributed nonconvex constrained optimization over time-varying digraphs. Mathematical Programming, 176(1):497–544, 2019.

Z. Song, L. Shi, S. Pu, and M. Yan. Optimal gradient tracking for decentralized optimization. arXiv preprint arXiv:2110.05282, 2021.

S. U. Stich. Local sgd converges fast and communicates little. arXiv preprint arXiv:1805.09767, 2018.

Y. Sun, G. Scutari, and A. Daneshmand. Distributed optimization based on gradient tracking revisited: Enhancing convergence rate via surrogation. SIAM Journal on Optimization, 32(2):354–385, 2022.

Y. Sun, G. Scutari, and D. Palomar. Distributed nonconvex multiagent optimization over time-varying networks. In 2016 50th Asilomar Conference on Signals, Systems and Computers, pages 788–794. IEEE, 2016.

K. K. Thekumparampil, N. He, and S. Oh. Lifted primal-dual method for bilinearly coupled smooth minmax optimization. arXiv preprint arXiv:2201.07427, 2022.

Y. Tian, G. Scutari, T. Cao, and A. Gasnikov. Acceleration in distributed optimization under similarity. In International Conference on Artificial Intelligence and Statistics, pages 5721–5756. PMLR, 2022.

E. Trimbach and A. Rogozin. An acceleration of decentralized sgd under general assumptions with low stochastic noise. In International Conference on Mathematical Optimization Theory and Operations Research, pages 117–128. Springer, 2021.

J. N. Tsitsiklis. Problems in decentralized decision making and computation. Technical report, Massachusetts Inst of Tech Cambridge Lab for Information and Decision Systems, 1984.

B. Woodworth, K. K. Patel, S. Stich, Z. Dai, B. Bullins, B. McMahan, O. Shamir, and N. Srebro. Is local sgd better than minibatch sgd? In International Conference on Machine Learning, pages 10334–10343. PMLR, 2020.

X. Wu and J. Lu. Fenchel dual gradient methods for distributed convex optimization over time-varying networks. IEEE Transactions on Automatic Control, 64(11):4629–4636, 2019.

L. Xiao, S. Boyd, and S.-J. Kim. Distributed average consensus with least-mean-square deviation. Journal of parallel and distributed computing, 67(1):33–46, 2007.