# Asynchronous Decentralized Optimization With Implicit Stochastic Variance Reduction

**Kenta Niwa** [1 2] **Guoqiang Zhang** [3] **W. Bastiaan Kleijn** [4]
**Noboru Harada** [1 2] **Hiroshi Sawada** [1] **Akinori Fujino** [1]

## Abstract

A novel asynchronous decentralized optimization method that follows Stochastic Variance Reduction (SVR) is proposed. Average consensus algorithms, such as Decentralized Stochastic Gradient Descent (DSGD), facilitate distributed training of machine learning models. However, the gradient will *drift* within the local nodes due to statistical heterogeneity of the subsets of data residing on the nodes and long communication intervals. To overcome the drift problem, (i) Gradient Tracking-SVR (GT-SVR) integrates SVR into DSGD and (ii) Edge-Consensus Learning (ECL) solves a model constrained minimization problem using a primal-dual formalism. In this paper, we reformulate the update procedure of ECL such that it implicitly includes the gradient modification of SVR by optimally selecting a constraint-strength control parameter. Through convergence analysis and experiments, we confirmed that the proposed ECL with Implicit SVR (ECL-ISVR) is stable and approximately reaches the reference performance obtained with computation on a single-node using full data set.

## 1. Introduction

While the use of massive data benefits the training of machine learning (ML) models, aggregating all data into one physical location (e.g., cloud data center) may overwhelm available communication bandwidth and violate rules on consumer privacy. In the European Union's General Data Protection Regulation (GDPR) (Custers et al., 2019) (article 46), a controller or processor may transfer personal data to a third country or an international organization only if the controller or processor has provided appropriate safeguards. Our goal is to facilitate ML model training without revealing the original data from local nodes. This requires edge computing (e.g., (Shi et al., 2016; Mao et al., 2017; Zhou et al., 2019)), which brings computation and data storage closer to the location where it is needed.

A representative collaborative learning algorithm is FedAvg (McMahan et al., 2017) for centralized networks. In FedAvg, the model update differences on a subset of local nodes are synchronously transmitted to a central server where they are averaged. *Average consensus* algorithms, such as DSGD (Chen & Sayed, 2012; Kar & Moura, 2013; Ram et al., 2010), Gossip SGD (GoSGD) (Ormándi et al., 2013; Jin et al., 2016; Blot et al., 2016), and related parallel algorithms (Sattler et al., 2019; Lim et al., 2020; Xie et al., 2019; Jiang et al., 2017; Lian et al., 2017; Tang et al., 2018) have been studied. However, it has been found empirically that these average consensus algorithms (even with model normalization (Li et al., 2019)) do not perform well when (i) the data subsets held on the local nodes are statistically *heterogeneous*,[1] (ii) use asynchronous and/or sparse (long update intervals) communication between local nodes, and (iii) the networks have arbitrary/non-homogeneous configurations.

In such scenarios, the gradient used for model update will often *drift* within the local nodes, resulting in either slow convergence or unstable iterates. An effective approach to overcome this issue is introduced in SCAFFOLD (Karimireddy et al., 2020), where SVR is applied to FedAvg. Later, the gradient control rule of SVR was applied to DSGD in GT-SVR (Xin et al., 2020). In GT-SVR, each local-node gradient bias is modified using expectations of both the global and local gradients (*control variates*) for each update iteration. Representative methods to calculate control variates are Stochastic Variance Reduced Gradient descent (SVRG) (Johnson & Zhang, 2013) and SAGA (Defazio et al., 2014). It is straightforward to include SVR in various algorithms, as externally calculated control variates are just added to the local stochastic gradient. However, there are uncertainties in the implementation of the control variable calculation. For example, SVRG updates the control variables by using first order local node gradients for each regular update interval, while a different update timing was used in SAGA.

[1] NTT Communication Science Laboratories, Kyoto, Japan [2] NTT Media Intelligence Laboratories, Tokyo, Japan [3] University of Technology Sydney, Sydney, Australia [4] Victoria University of Wellington, Wellington, New Zealand. Correspondence to: Kenta Niwa <kenta.niwa.bk@hco.ntt.co.jp>.

*Proceedings of the $38^{th}$ International Conference on Machine Learning*, PMLR 139, 2021. Copyright 2021 by the author(s).

[1] This class of problems has also been referred to as ”non-IID”.

Another important approach towards addressing the gradient drift issue is to solve a linearly constrained cost function minimization problem to make the model variables identical among the nodes. A basic solver for this problem applies a *primal-dual formalism* using a Lagrangian function. Representative algorithms on decentralized networks are the Primal-Dual Method of Multipliers (PDMM) (Zhang & Heusdens, 2017; Sherson et al., 2018) and its extension to non-convex DNN optimization named Edge-Consensus Learning (ECL) (Niwa et al., 2020). When applying PDMM to centralized networks, it reduces to the recently developed FedSplit method (Pathak & Wainwright, 2020). By using the primal-dual formalism, the gradient modification terms result naturally from the dual variables associated with the model constraints. However, without a careful parameter search to control the constraint strength, this approach may not be effective in preventing gradient drift.

SVR may be applicable even for the primal-dual formalism. In fact, it was recently reported in (Rajawat & Kumar, 2020) that externally calculated control variates using, e.g., SVRG or SAGA can be added to the stochastic gradient of the update procedure of PDMM. It is natural to assume that the primal-dual formalism and SVR are not independent, but linked because both approaches are expected to be effective in terms of gradient drift reduction. Hence it is desirable to develop an algorithm that simultaneously takes advantage of both approaches when computing the gradient control variates instead of adding SVR externally to an existing method as (Rajawat & Kumar, 2020). Thus, we propose to derive the control variates by solving a model constrained minimization problem with a primal-dual formalism. Since the constraint strength control parameter is included in a primal-dual formalism, its optimal selection is natural. Thus, our novel approach represents an advance over many SVR studies for decentralized optimization as it eliminates the implementational ambiguity of the control variates.

In this paper, we propose ECL-ISVR, which reformulates the primal-dual formalism algorithm (ECL) such that it implicitly includes the gradient control variates of SVR (see Sec. 5). By inspection of the physical meaning of e.g., dual variables in ECL, we noticed that they are proportional to the local gradient expectation, which are components of the gradient control variates of SVR. By optimally selecting the constraint strength control parameter to scale dual variables, the update procedure matches that of SVR. By doing so, we avoid the implementational ambiguity of the control variates of SVR because they are implicitly updated in the primal-dual formalism algorithm. Since it eliminates additional external operations as in (Rajawat & Kumar, 2020), the proposed algorithm is expected to have small update errors and to be robust in practical scenarios. We provide a convergence analysis for ECL-ISVR for the strongly convex, general convex, and non-convex cases in Sec. 6. We evaluate and confirm the advantages of the proposed algorithms using benchmark image classification problems with both convex and non-convex cost functions (Sec. 7).

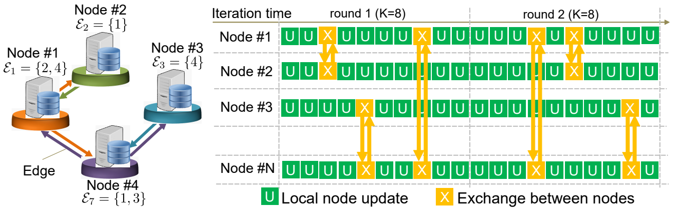

*Figure 1.* Decentralized network with asynchronous arbitrary node connections and example procedure/communication schedule.

## 2. Preliminary steps

We define symbols and notation in subsection 2.1. In subsection 2.2, definitions that are needed for the convergence analysis are summarized.

### 2.1. Network settings and symbols

We now introduce the symbols and notations used throughout the paper. Let us consider a decentralized network in Fig. 1, a set $\mathcal{N}$ of $N$ local nodes is connected with arbitrary graphical structure $\mathcal{G}(\mathcal{N}, \mathcal{E})$, where $\mathcal{E}$ is a set of $E$ bidirectional edges. A local node communicates with only a small number of fixed nodes in an asynchronous manner. We denote set cardinality by $|\cdot|$, so that $N = |\mathcal{N}|$ and $E = |\mathcal{E}|$. The index set of neighbors connected to the $i$-th node is written as $\mathcal{E}_i = \{j \in \mathcal{N} | (i, j) \in \mathcal{E}\}$. For counting the number of edges, we use $E_i = |\mathcal{E}_i|$, implying $E = \sum_{i \in \mathcal{N}} E_i$. The data subsets $\boldsymbol{x}_i$ are sets of cardinality $|\boldsymbol{x}_i|$ containing $\zeta$-dimensional samples available for each local node $i \in \mathcal{N}$. The $\boldsymbol{x}_i$ may be heterogeneous, which implies that $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ $(i \neq j)$ are sampled from different distributions. A communication schedule example is shown in Fig. 1. Assuming that the computational/communication performance of all local nodes is similar, $K$ updates will be performed at each local node for each node communication round $r \in \{1, ..., R\}$, namely each round contains $K$ inner iterations for each edge. Each node communicates once per round and the set of neighbor nodes connected with node $i$ at iteration $(r, k)$ is denoted by $\mathcal{E}_i^{r,k}$.

Our distributed optimization procedure can be applied to any ML model. Prior to starting the training procedure, identical model architectures and local cost functions $f_i$ are defined for all nodes $(f_i = f_j | i, j \in \mathcal{N})$. The cost function at local node $i$ is $f_i(\boldsymbol{w}_i) = \mathbb{E}_{\boldsymbol{\chi}_i \sim \boldsymbol{x}_i}[f_i(\boldsymbol{w}_i; \boldsymbol{\chi}_i)]$, where $\boldsymbol{w}_i \in \mathbb{R}^m$ represents the model variables at the $i$-th local node and $\boldsymbol{\chi}_i$ denotes a mini-batch data sample from $\boldsymbol{x}_i$. The cost function $f_i : \mathbb{R}^\zeta \to \mathbb{R}$ is assumed to be Lipschitz smooth and it can be convex or non-convex (e.g., DNN). Thus, $f_i$ is differentiable and the stochastic gradient is calculated as $g_i(\boldsymbol{w}_i) = \nabla f_i(\boldsymbol{w}_i; \boldsymbol{\chi}_i)$, where $\nabla$ denotes the differential operator. Our goal is to find the model variables that minimize the global cost function $f(\boldsymbol{w}) = \frac{1}{N}\sum_{i\in\mathcal{N}} f_i(\boldsymbol{w}_i)$ while making the local node models identical as much as possible ($\boldsymbol{w}_i = \boldsymbol{w}_j$), where the stacked model variables are given by $\boldsymbol{w} = [\boldsymbol{w}_1^{\mathrm{T}}, ..., \boldsymbol{w}_N^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{Nm}$ and $^{\mathrm{T}}$ denotes transpose.

### 2.2. Definitions

Next, we list several definitions that are used in the theoretical convergence analysis.

(D1) $\beta$**-Lipschitz smooth**: When $f_i$ is assumed to be Lipschitz smooth, there exists a constant $\beta \in [0, \infty]$ satisfying

$$\|\nabla f_i(\boldsymbol{w}_i) - \nabla f_i(\boldsymbol{u}_i)\| \leq \beta \|\boldsymbol{w}_i - \boldsymbol{u}_i\|, \quad (\forall i, \boldsymbol{w}_i, \boldsymbol{u}_i).$$

This assumption implies the following inequality:

$$f_i(\boldsymbol{u}_i) \leq f_i(\boldsymbol{w}_i) + \langle \nabla f_i(\boldsymbol{w}_i), \boldsymbol{u}_i - \boldsymbol{w}_i \rangle + \tfrac{\beta}{2}\|\boldsymbol{u}_i - \boldsymbol{w}_i\|^2.$$

(D2) $\alpha$**-convex**: When $f_i$ is assumed to be convex, there exists a constant $\alpha \in (0, \beta)$ satisfying for any two points $\{\boldsymbol{w}_i, \boldsymbol{u}_i\}$,

$$f_i(\boldsymbol{u}_i) \geq f_i(\boldsymbol{w}_i) + \langle \nabla f_i(\boldsymbol{w}_i), \boldsymbol{u}_i - \boldsymbol{w}_i \rangle + \tfrac{\alpha}{2}\|\boldsymbol{u}_i - \boldsymbol{w}_i\|^2.$$

Although $\alpha = 0$ is allowed for *general convex* cases, $\alpha > 0$ is guaranteed for *strongly convex* cases.

(D3) $\sigma^2$**-bounded variance**: $g_i(\boldsymbol{w}_i) = \nabla f_i(\boldsymbol{w}_i; \boldsymbol{\chi}_i)$ is an unbiased stochastic gradient of $f_i$ with bounded variance,

$$\mathbb{E}[\|g_i(\boldsymbol{w}_i) - \nabla f_i(\boldsymbol{w}_i)\|^2] \leq \sigma^2 \quad (\forall i, \boldsymbol{w}_i).$$

## 3. Average consensus and its SVR application

Average consensus algorithms, such as DSGD, aim to solve a simple cost minimization problem:

$$\inf_{\boldsymbol{w}} f(\boldsymbol{w}). \tag{1}$$

The algorithms follow model averaging with a local node update based on stochastic gradient descent (SGD) using the forward Euler method with step-size $\mu$ $(>0)$ as $\boldsymbol{w}_i^{r,k+1} = \boldsymbol{w}_i^{r,k} - \mu g_i(\boldsymbol{w}_i^{r,k})$, where $r$ labels the update *round* and $k$ denotes inner iteration. The forward update procedure is decomposed into $K$ updates of $\boldsymbol{w}_i$ performed on $N$ local nodes. In e.g., FedProx (Li et al., 2019), a normalization term to make the $N$ node model variables be closer with weight $\upsilon$ $(>0)$ is added to the cost function as

$$\inf_{\boldsymbol{w}} f(\boldsymbol{w}) + \tfrac{\upsilon}{2}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{E}_i}\|\boldsymbol{w}_i - \boldsymbol{w}_j\|^2. \tag{2}$$

It has been reported that in heterogeneous data settings and/or long communication intervals, ($K \neq 1$), so-called *gradient drift* commonly occurs in average consensus algorithms. An approach that aims to address this issue is to apply an SVR method to DSGD, such as SVRG (Johnson & Zhang, 2013) and SAGA (Defazio et al., 2014). A modified gradient $\bar{g}_i(\boldsymbol{w}_i)$ is obtained by using global $\bar{\boldsymbol{c}}_i$ and local control variate $\boldsymbol{c}_i$, respectively, for correcting the gradient on the $i$-th node as

$$\bar{g}_i(\boldsymbol{w}_i) \leftarrow g_i(\boldsymbol{w}_i) + \bar{\boldsymbol{c}}_i - \boldsymbol{c}_i, \tag{3}$$

where the modified gradient is unbiased as the global variate is $\bar{\boldsymbol{c}}_i = \mathbb{E}_{j,r,k}[g_j(\boldsymbol{w}_i)]$ and the local control variate is $\boldsymbol{c}_i = \mathbb{E}_{r,k}[g_i(\boldsymbol{w}_i)]$, where $\mathbb{E}_{j,r,k}$ and $\mathbb{E}_{r,k}$ denote expectation w.r.t. both nodes connected with $i$-th node ($j \in \mathcal{N}_i$) and time $(r, k)$ and that w.r.t. time, respectively. It immediately follows that $\mathbb{E}_{j,r,k}[\bar{g}_i(\boldsymbol{w}_i)] = \mathbb{E}_{r,k}[g_i(\boldsymbol{w}_i)]$. The advantage of the approach is that the variance of $\bar{g}_i(\boldsymbol{w}_i)$ is guaranteed to be lower than that of $g_i(\boldsymbol{w}_i)$ if $\mathrm{Var}[\boldsymbol{c}_i] \leq 2\mathrm{Cov}[g_i(\boldsymbol{w}_i), \boldsymbol{c}_i]$. GT-SVR (Xin et al., 2020) integrates SVR techniques into DSGD for a fully-connected decentralized network.

It is discussed above that the gradient modification of (3) results in gradient variance reduction. However, the implementation of the control variates $\{\bar{\boldsymbol{c}}_i, \boldsymbol{c}_i\}$ is ambiguous. For a better implementation, the modification must be formulated as the result of a change in the cost function. Based on such a cost function perspective, we provide mathematically rigorous derivations of distributed consensus algorithms that facilitate further extension in the future.

## 4. Primal-dual formalism

### 4.1. Problem definition

DSGD solves the decentralized model learning problem using a straightforward cost minimization approach (1). Instead, we reformulate this problem as a more general linearly constrained minimization problem that is more effective in making the model variables identical across the nodes by reducing gradient drift:

$$\inf_{\boldsymbol{w}} f(\boldsymbol{w}) \text{ s.t. } \mathbf{A}_{i|j}\boldsymbol{w}_i + \mathbf{A}_{j|i}\boldsymbol{w}_j = \mathbf{0}, \quad (\forall i \in \mathcal{N}, j \in \mathcal{E}_i), \tag{4}$$

where $\mathbf{A}_{i|j} \in \mathbb{R}^{m\times m}$ is the constraint parameter for the edge $(i, j)$ at node $i$. Since the set of $\mathbf{A}_{i|j}$ must force the model variables to be identical over all nodes, we use identity matrices with opposite signs, $\{\mathbf{A}_{i|j}, \mathbf{A}_{j|i}\} = \{\mathbf{I}, -\mathbf{I}\}$.

The linearly constrained minimization problem for convex cost function can be solved using a primal-dual formalism (Fenchel, 1949). An effective approach to solve (4) on an asynchronous decentralized network is PDMM (Zhang & Heusdens, 2017; Sherson et al., 2018) and its extension to optimize non-convex DNN models, ECL (Niwa et al., 2020). Then, $f_i$ is assumed to be $\beta$-Lipschitz smooth but not necessarily convex, and it is natural to solve the majorization minimization of $f_i$ by defining a locally quadratic function $q_i$ around $\boldsymbol{w}_i^{r,k}$,

$$q_i(\boldsymbol{w}_i) = f_i(\boldsymbol{w}_i^{r,k}) + \langle g_i(\boldsymbol{w}_i^{r,k}), \boldsymbol{w}_i - \boldsymbol{w}_i^{r,k} \rangle + \tfrac{1}{2\mu}\|\boldsymbol{w}_i - \boldsymbol{w}_i^{r,k}\|^2.$$

When the step-size $\mu$ is sufficiently small, $\mu \leq 1/\beta$, then $q_i(\boldsymbol{w}_i) \geq f_i(\boldsymbol{w}_i)$ is guaranteed everywhere. By replacing $f(\boldsymbol{w})$ in (4) by $q(\boldsymbol{w}) = \frac{1}{N}\sum_{i\in\mathcal{N}} q_i(\boldsymbol{w}_i)$, a dual problem can be defined.

To formulate the dual problem, we first define some variables. Let $\mathbf{A} = \mathrm{Diag}\{[\mathbf{A}_1, ..., \mathbf{A}_N]\}$ be a block diagonal matrix with the $\mathbf{A}_i = \mathrm{Diag}\{[\mathbf{A}_{i|\mathcal{E}_i(1)}, ..., \mathbf{A}_{i|\mathcal{E}_i(E_i)}]\} \in \mathbb{R}^{E_i m \times E_i m}$ associated with the linear constraints. We will use *lifted* dual variables for controlling the constraint strength to facilitate asynchronous node communication. The lifted dual variables are written as $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1^{\mathrm{T}}, \boldsymbol{\lambda}_2^{\mathrm{T}}, ..., \boldsymbol{\lambda}_N^{\mathrm{T}}]^{\mathrm{T}}$, with each lifted dual variable element held at a node relating to one of its neighbours, $\boldsymbol{\lambda}_i = [\boldsymbol{\lambda}_{i|\mathcal{E}_i(1)}^{\mathrm{T}}, ..., \boldsymbol{\lambda}_{i|\mathcal{E}_i(E_i)}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{E_i m}$. We will use an indicator function to enforce the equality of the lifted variables, $\boldsymbol{\lambda}_{i|j} = \boldsymbol{\lambda}_{j|i}$. For this purpose, we define $\iota_{\ker(b)}$ as the indicator function that takes the value 0 for $b = 0$ and is $\infty$ elsewhere. Finally, let $q^\star$ be the convex conjugate of $q$. Then, the dual problem of (4) can be formulated as (refer to (Niwa et al., 2020) for more detail):

$$\inf_{\boldsymbol{\lambda}} q^\star(\mathbf{J}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\boldsymbol{\lambda}) + \iota_{\ker(\mathbf{I}-\mathbf{P})}(\boldsymbol{\lambda}), \tag{5}$$

where the mixing matrix $\mathbf{J}$ connects $\boldsymbol{w}$ and $\boldsymbol{\lambda}$ and is given by $\mathbf{J} = \mathrm{Diag}\{[\mathbf{J}_1, ..., \mathbf{J}_N]\}$ composed of $\mathbf{J}_i = [\mathbf{I}, ..., \mathbf{I}]^{\mathrm{T}} \in \mathbb{R}^{E_i m \times m}$ and where $\mathbf{P} \in \mathbb{R}^{Em \times Em}$ denotes the permutation matrix that exchanges the lifted dual variables between connected nodes as $\boldsymbol{\lambda}_{i|j} \rightleftharpoons \boldsymbol{\lambda}_{j|i} (\forall i \in \mathcal{N}, j \in \mathcal{E}_i)$. We note that the convex conjugate function is $q^\star(\mathbf{J}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\boldsymbol{\lambda}) = \sup_{\boldsymbol{w}}\big(\langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{J}\boldsymbol{w}\rangle - q(\boldsymbol{w})\big)$. The indicator function that enforces equality on the lifted dual variables takes the value zero only when $(\mathbf{I}-\mathbf{P})\boldsymbol{\lambda} = \mathbf{0}$ is satisfied.

## 4.2. Update rules of ECL

We now briefly review the update rules of ECL, which aim to solve (5). Since the two cost terms in (5) are ill-matched because the indicator function is nondifferentiable, it is difficult to reduce overall cost by, e.g., subtracting a scaled cost gradient. In such situations, applying operator splitting e.g., (Bauschke et al., 2011; Ryu & Boyd, 2016) can be effective. The existing form of ECL allows two flavors of operator splitting, Peaceman-Rachford Splitting (PRS) (Peaceman & Rachford, 1955) and Douglas-Rachford Splitting (DRS) (Douglas & Rachford, 1956), to obtain recursive variable update rules associated with PDMM (Zhang & Heusdens, 2017; Sherson et al., 2018) and ADMM (Gabay & Mercier, 1976), respectively. In ECL, the augmented Lagrangian problem, which adds a model proximity term $\frac{\rho}{2}\|\mathbf{J}\boldsymbol{w} - \mathbf{P}\mathbf{J}\boldsymbol{w}^{r,k}\|^2$ $(\rho \geq 0)$ to $q_i$, is solved. The resulting alternating update rules of the primal variable $\boldsymbol{w}$ and auxiliary lifted dual variables $\{\boldsymbol{y}, \boldsymbol{z}\}$ constitute PDMM-SGD (PRS) and ADMM-SGD (DRS):

$$\begin{aligned}\boldsymbol{w}^{r,k+1} &= \arg\min_{\boldsymbol{w}}\big(q(\boldsymbol{w}) \\ &\quad + \tfrac{\eta}{2}\|\mathbf{A}\mathbf{J}\boldsymbol{w} - \boldsymbol{z}^{r,k}\|^2 + \tfrac{\rho}{2}\|\mathbf{J}\boldsymbol{w} - \mathbf{P}\mathbf{J}\boldsymbol{w}^{r,k}\|^2\big),\end{aligned} \tag{6}$$

$$\boldsymbol{y}^{r,k+1} = \boldsymbol{z}^{r,k} - 2\mathbf{A}\mathbf{J}\boldsymbol{w}^{r,k+1}, \tag{7}$$

$$\boldsymbol{z}^{r,k+1} = \begin{cases}\mathbf{P}\boldsymbol{y}^{r,k+1} & \text{(PDMM-SGD)} \\ \frac{1}{2}\mathbf{P}\boldsymbol{y}^{r,k+1} + \frac{1}{2}\boldsymbol{z}^{r,k} & \text{(ADMM-SGD)}\end{cases}, \tag{8}$$

**Algorithm 1** Previous ECL (Niwa et al., 2020)

```
▷ Set $\boldsymbol{w}_i = \boldsymbol{w}_j = \boldsymbol{u}_{i|j} (\sim \text{Norm}), \boldsymbol{z}_{i|j} = \mathbf{0}, f_i, \mu, \eta_i, \rho_i,$
    $\boldsymbol{x}_i, \mathbf{A}_{i|j}$
for $r \in \{1, \dots, R\}$ (Outer loop round) do
  for $i \in \mathcal{N}$ do
    for $k \in \{1, \dots, K\}$ (Inner loop iteration) do
      ▷ Stochastic gradient calculation
      $g_i(\boldsymbol{w}_i) \leftarrow \nabla f_i(\boldsymbol{w}_i, \boldsymbol{\chi}_i)$
      ▷ Update local primal and lifted dual variables
      $\boldsymbol{w}_i \leftarrow \{\boldsymbol{w}_i - \mu g_i(\boldsymbol{w}_i) + \mu \sum_{j \in \mathcal{E}_i} (\eta_i \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}$
        $+ \rho_i \boldsymbol{u}_{i|j})\} / (1 + \mu E_i(\eta_i + \rho_i))$
      for $j \in \mathcal{E}_i$ do
        $\boldsymbol{y}_{i|j} \leftarrow \boldsymbol{z}_{i|j} - 2\mathbf{A}_{i|j}\boldsymbol{w}_i$
      end for
      ▷ Procedure when communicated with $j$-th node
      for $j \in \mathcal{E}_i^{r,k}$ (at random time) do
        communicate$_{j \to i}(\boldsymbol{w}_j, \boldsymbol{y}_{j|i})$
        $\boldsymbol{u}_{i|j} \leftarrow \boldsymbol{w}_j$
        $\boldsymbol{z}_{i|j} \leftarrow \begin{cases} \boldsymbol{y}_{j|i} & \text{(PDMM-SGD)} \\ \frac{1}{2}\boldsymbol{z}_{i|j} + \frac{1}{2}\boldsymbol{y}_{j|i} & \text{(ADMM-SGD)} \end{cases}$
      end for
    end for
  end for
end for
```

where the penalty term $\frac{\eta}{2}\|\mathbf{A}\mathbf{J}\boldsymbol{w} - \boldsymbol{z}^{r,k}\|^2$ $(\eta > 0)$ in (6) results from the linear constraints in (4). It reduces the gradient drift for each local node, so that the model variables over the $N$ nodes are close. The update rules (6)–(8) can be decomposed into procedures on the local nodes, as summarized in Alg. 1. A pair of a primal model variable and a dual variable $\{\boldsymbol{w}_j, \boldsymbol{y}_{j|i}\}$ or their update differences are transmitted between nodes according to the communication schedule of Fig. 1. Since DRS uses averaging with the previous value, the computation memory of ADMM-SGD (DRS) is larger than that of PDMM-SGD (PRS).

In PDMM and ECL, the following problems (P1), (P2) remain unsolved:

(P1) Non-optimal $\eta_i$: Since $\eta_i$ is associated with constrained force strength, it is empirically scaled such that it is inversely proportional to the parameters, i.e., the number of model element $\boldsymbol{e}$ as $\eta_i \propto 1/\sqrt{\boldsymbol{e}}$, similar to model initialization of (He et al., 2015). In our pre-testing using open source[2], ECL requires careful selection of $\eta_i$ to prevent gradient drift.

(P2) Doubled communication requirement: In ECL, a pair of primal and dual variables $\{\boldsymbol{w}_j, \boldsymbol{y}_{j|i}\}$ are transmitted, whereas DSGD exchanges only $\boldsymbol{w}_j$. When the variable update dynamics is stable, ignoring the proximity term $(\rho_i = 0)$ may be allowed. Then, ECL transmits only $\boldsymbol{y}_{j|i}$, where its variable dimension is identical to that of $\boldsymbol{w}_j$.

[2] https://github.com/nttcslab/edge-consensus-learning

## 5. Proposed algorithms (ECL-ISVR)

As noted in Sec. 1, it is natural to assume that the primal-dual formalism (ECL) and SVR must be related because both approaches aim to reduce gradient drift. More specifically, the $\boldsymbol{w}_i$-update procedure in Alg. 1 is similar to the gradient modification of SVR in (3), as $\eta_i \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}$ is added to the stochastic gradient. If the gradient control variates of SVR can be represented by means of dual variables with optimal $\eta_i$-selection in ECL, we can eliminate the ambiguities in control variate implementation of SVR. Hence, the aim of this section is to derive the SVR gradient modification (3) by simply reformulating the update procedure in Alg. 1.

In Subsec. 5.1, we start with investigating the physical meaning of various terms (e.g., $\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}$) and reformulate the $\boldsymbol{w}_i$-update procedure to match that of SVR. This will include optimal $\eta_i$-selection, thus solving (P1). After reformulation, the $\boldsymbol{w}_i$-update procedure follows SVR, achieving a stable variable update dynamics and additional model normalization will be unnecessary, ($\rho_i = 0$), thus halving the communication requirement and solving (P2). Our new algorithms, ECL-ISVR composed of PDMM-ISVR (PRS) and ADMM-ISVR (DRS), are summarized in Subsec. 5.2.

### 5.1. The correspondence between ECL and SVR

We first discuss some preliminaries needed for investigating the terms in Alg. 1. The permutation matrix satisfies $\mathbf{PP} = \mathbf{I}$, furthermore $\mathbf{A}^{\mathrm{T}}\mathbf{A} = \mathbf{I}$, and $\mathbf{PAP} = -\mathbf{A}$ because $\{\mathbf{A}_{i|j}, \mathbf{A}_{j|i}\} = \{\mathbf{I}, -\mathbf{I}\}$. The mixing matrix satifies $\mathbf{J}^{\mathrm{T}}\mathbf{J} = \mathrm{Diag}\{[E_1\mathbf{I}, ..., E_N\mathbf{I}]\}$.

To model the update lag of the variables resulting from asynchronous communication (variables are exchanged once per $K$ inner iterations with random timing for each edge as shown in Fig. 1) we define an additional variable to keep track of the updates through the individual edges for the round as $\boldsymbol{u}_{i|j}^{r} = \boldsymbol{w}_i^{r,\kappa(i,j,r)} \in \mathbb{R}^m$, where $\kappa(i,j,r)$ denotes the inner iteration index for communicating from the $i$-th node to the $j$-th in round $r$. Since the dual variables are associated with the primal variables as in (7) and transmitted between nodes by (8), it is natural to represent $\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j} \in \mathbb{R}^m$ considering the update lag by using $\boldsymbol{u}_{i|j}^{r}$. This variable can be stacked as $\boldsymbol{u} = [\boldsymbol{u}_1^{\mathrm{T}}, ..., \boldsymbol{u}_N^{\mathrm{T}}] \in \mathbb{R}^{Em}$ composed of $\boldsymbol{u}_i = [\boldsymbol{u}_{i|\mathcal{E}_i(1)}^{\mathrm{T}}, ..., \boldsymbol{u}_{i|\mathcal{E}_i(E_i)}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{E_i m}$.

To investigate the physical meaning of $\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}$ in Alg. 1, we substitute the PRS update procedure (7) into (8) with initialization $\boldsymbol{z}^{1,0} = \mathbf{0}$, with the replacement of $\mathbf{J}\boldsymbol{w}$ in (7) by $\boldsymbol{u}$ to model communication lags. The result is different when $r$ is an odd number and when it is an even number. When $r$ is an odd number, it results in

$$\begin{aligned}\mathbf{A}^{\mathrm{T}}\boldsymbol{z}^{r,0} &= \mathbf{A}^{\mathrm{T}}\mathbf{P}(\boldsymbol{z}^{r-1,0} - 2\mathbf{A}\boldsymbol{u}^{r-1}) \\ &= \mathbf{A}^{\mathrm{T}}\mathbf{P}(\boldsymbol{z}^{r-2,0} - 2\mathbf{A}\boldsymbol{u}^{r-2}) - 2\mathbf{A}^{\mathrm{T}}\mathbf{PA}\boldsymbol{u}^{r-1} \\ &= \mathbf{A}^{\mathrm{T}}\mathbf{P}\boldsymbol{z}^{1,0} - 2\textstyle\sum_{l=1}^{(r-1)/2}\mathbf{A}^{\mathrm{T}}(\mathbf{PAPP}\boldsymbol{u}^{2l} + \mathbf{A}\boldsymbol{u}^{2l-1}) \\ &= 2\textstyle\sum_{l=1}^{(r-1)/2}(\mathbf{P}\boldsymbol{u}^{2l} - \boldsymbol{u}^{2l-1}) \quad \text{(PRS)}. \qquad (9)\end{aligned}$$

In the next round ($r+1$ is then even), the update is

$$\mathbf{A}^{\mathrm{T}}\boldsymbol{z}^{r+1,0} = 2\textstyle\sum_{l=1}^{(r-1)/2}(\mathbf{P}\boldsymbol{u}^{2l+1} - \boldsymbol{u}^{2l}) + 2\mathbf{P}\boldsymbol{u}^{r} \ \text{(PRS)}. \ (10)$$

Remembering that the $u_{i|j}$ represents the model variables, (9) and (10) show that $\mathbf{A}^{\mathrm{T}}\boldsymbol{z}$ in PRS is the cumulative sum of model differences between nodes and it is updated every two rounds.

We now compare the results (9) and (10) to SVRG and SAGA. There similar model differences between nodes with appropriate scaling are used to represent the global control variate $\bar{\boldsymbol{c}}_i = \mathbb{E}_{j,r,k}[g_j(\boldsymbol{w}_i)]$ $(j \in \mathcal{E}_i)$. To cancel out the effect of the step-size $\mu$, the number of the inner loop iteration $K$, and that of connected nodes $E_i$, multiplying $\sum_{j\in\mathcal{E}_i} \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}$ with $-1/(\mu K E_i)$ is the optimal choice for PRS. Assuming that $r$ is an odd number, this results in

$$\begin{aligned}&-\tfrac{1}{\mu K E_i}\textstyle\sum_{j\in\mathcal{E}_i}\mathbf{A}_{i|j}^{\mathrm{T}}\boldsymbol{z}_{i|j}^{r,0} \\ &= \tfrac{1}{\mu K E_i}\textstyle\sum_{j\in\mathcal{E}_i}\sum_{l=1}^{(r-1)/2} 2(\boldsymbol{u}_{i|j}^{2l-1} - \boldsymbol{u}_{j|i}^{2l}) \ \text{(PRS)}. \quad (11)\end{aligned}$$

Similarly to PRS, substituting (7) into the update rule of DRS (8) with initialization $\boldsymbol{z}^{1,0} = \mathbf{0}$ results in

$$\begin{aligned}&\mathbf{A}^{\mathrm{T}}\boldsymbol{z}^{r+1,0} = \tfrac{1}{2}\mathbf{A}^{\mathrm{T}}\mathbf{P}(\boldsymbol{z}^{r,0} - 2\mathbf{A}\boldsymbol{u}^{r}) + \tfrac{1}{2}\mathbf{A}^{\mathrm{T}}\boldsymbol{z}^{r,0} \\ &= \tfrac{1}{2}\mathbf{A}^{\mathrm{T}}(\mathbf{I}+\mathbf{P})\boldsymbol{z}^{r,0} - \tfrac{1}{2}\mathbf{A}^{\mathrm{T}}(\mathbf{PAPP}+\mathbf{A})\boldsymbol{u}^{r-1} - \mathbf{A}^{\mathrm{T}}\mathbf{PAPP}\boldsymbol{u}^{r} \\ &= \tfrac{1}{2}\mathbf{A}^{\mathrm{T}}(\mathbf{I}+\mathbf{P})\boldsymbol{z}^{1,0} + \tfrac{1}{2}\textstyle\sum_{l=1}^{r-1}(\mathbf{P} - \mathbf{I})\boldsymbol{u}^{l} + \mathbf{P}\boldsymbol{u}^{r} \\ &= \tfrac{1}{2}\textstyle\sum_{l=1}^{r-1}(\mathbf{P} - \mathbf{I})\boldsymbol{u}^{l} + \mathbf{P}\boldsymbol{u}^{r} \qquad \text{(DRS)}, \qquad (12)\end{aligned}$$

for any $r$. Although an offset $\mathbf{P}\boldsymbol{u}^{r}$ is added in (12), this can be ignored as it is just a fraction that arises in the recursive formulation of model differences between nodes. It is appropriate for DRS to select $-1/(\mu K E_i)$ as scaling factor. Then, multiplying with $\sum_{j\in\mathcal{E}_i}\mathbf{A}_{i|j}^{\mathrm{T}}\boldsymbol{z}_{i|j}$ results in

$$\begin{aligned}&-\tfrac{1}{\mu K E_i}\textstyle\sum_{j\in\mathcal{E}_i}\mathbf{A}_{i|j}^{\mathrm{T}}\boldsymbol{z}_{i|j}^{r+1,0} \\ &= \tfrac{1}{\mu K E_i}\textstyle\sum_{j\in\mathcal{E}_i}(\sum_{l=1}^{r-1}\tfrac{1}{2}(\boldsymbol{u}_{i|j}^{l} - \boldsymbol{u}_{j|i}^{l}) - \boldsymbol{u}_{j|i}^{r}) \ \text{(DRS)}. \ (13)\end{aligned}$$

From (11) and (13), it is found that $\sum_{j\in\mathcal{E}_i}\mathbf{A}_{i|j}^{\mathrm{T}}\boldsymbol{z}_{i|j}$ with appropriate scaling $1/(\mu K E_i)$ is associated with the global control variates $\bar{\boldsymbol{c}}_i$. However, difference between PRS and DRS exists, namely the primal model variables of the same round are used in DRS as $(\boldsymbol{u}_{i|j}^{l} - \boldsymbol{u}_{j|i}^{l})$, while those of a different round are used in PRS as $(\boldsymbol{u}_{i|j}^{2l-1} - \boldsymbol{u}_{j|i}^{2l})$. While the performance differences between PRS and DRS are investigated in the experiments in Sec. 7, we expect the convergence rate not to differ significantly since there are no essential differences in the physical meaning of the variables defined in (11) and those in (13).

Since the existing terms in Alg. 1 are associated with the global control variate, we would like to see if we can also identify the local control variate $\boldsymbol{c}_i = \mathbb{E}_{r,k}[g_i(\boldsymbol{w}_i)]$ within

**Algorithm 2** Proposed ECL-ISVR

```
▷ Set $\boldsymbol{w}_i = \boldsymbol{w}_j (\sim\mathrm{Norm}), \boldsymbol{z}_{i|j} = \boldsymbol{0}, \mu, \boldsymbol{x}_i, \mathbf{A}_{i|j}$
for $r \in \{1, \ldots, R\}$ (Outer loop round) do
    for $i \in \mathcal{N}$ do
        for $k \in \{1, \ldots, K\}$ (Inner loop iteration) do
            ▷ Stochastic gradient calculation
            $g_i(\boldsymbol{w}_i) \leftarrow \nabla f_i(\boldsymbol{w}_i, \boldsymbol{\chi}_i)$
            ▷ Update local primal and lifted dual variables
            $\boldsymbol{w}_i \leftarrow \boldsymbol{w}_i - \mu[g_i(\boldsymbol{w}_i^{r,k}) + \frac{1}{\mu K E_i}\{\sum_{j\in\mathcal{E}_i}$
                    $(\boldsymbol{w}_i^{r,k} - \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k}) - \mu E_i g_i(\boldsymbol{w}_i^{r,k})\}]$
            for $j \in \mathcal{E}_i$ do
                $\boldsymbol{y}_{i|j} \leftarrow \boldsymbol{z}_{i|j} - 2\mathbf{A}_{i|j}\boldsymbol{w}_i$
            end for
            ▷ Procedure when communicated with $j$-th node
            for $j \in \mathcal{E}_i^{r,k}$ (at random time) do
                communicate$_{j\rightarrow i}(\boldsymbol{y}_{j|i})$
                $\boldsymbol{z}_{i|j} \leftarrow \begin{cases} \boldsymbol{y}_{j|i} & \text{(PDMM-ISVR)} \\ \frac{1}{2}\boldsymbol{z}_{i|j} + \frac{1}{2}\boldsymbol{y}_{j|i} & \text{(ADMM-ISVR)} \end{cases}$
            end for
        end for
    end for
end for
```

ECL. To this purpose, let us reformulate the $\boldsymbol{w}_i$-update procedure in Alg. 1 such that it matches with SVR's update rule (3) with $\rho_i = 0$ to reduce the communication requirements:

$$
\begin{aligned}
&\boldsymbol{w}_i^{r,k+1} \\
&= (\boldsymbol{w}_i^{r,k} - \mu g_i(\boldsymbol{w}_i^{r,k}) + \mu\eta_i \textstyle\sum_{j\in\mathcal{E}_i} \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k})/(1+\mu\eta_i E_i) \quad (14)\\
&= (1 - \tfrac{\mu\eta_i E_i}{1+\mu\eta_i E_i})(\boldsymbol{w}_i^{r,k} - \mu g_i(\boldsymbol{w}_i^{r,k})) + \tfrac{\mu\eta_i}{1+\mu\eta_i E_i} \textstyle\sum_{j\in\mathcal{E}_i} \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k}\\
&= \boldsymbol{w}_i^{r,k} - \mu[g_i(\boldsymbol{w}_i^{r,k}) + \tfrac{\eta_i}{1+\mu\eta_i E_i}\{\textstyle\sum_{j\in\mathcal{E}_i}(\boldsymbol{w}_i^{r,k} - \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k})\\
&\qquad - \mu E_i g_i(\boldsymbol{w}_i^{r,k})\}]. \quad (15)
\end{aligned}
$$

Hereafter, we write $\Upsilon_i = 1/(1+\mu\eta_i E_i) \in [0, 1]$ to simplify notation. Since $\boldsymbol{w}_i$ is recursively updated following (14), the term $\eta_i \Upsilon_i \{\sum_{j\in\mathcal{E}_i}(\boldsymbol{w}_i - \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}) - \mu E_i g_i(\boldsymbol{w}_i)\}$ in (15) can be written as a summation of three terms $\mathcal{T}_1$, $\mathcal{T}_2$, and $\mathcal{T}_3$,

$$\eta_i \Upsilon_i \{\textstyle\sum_{j\in\mathcal{E}_i}(\boldsymbol{w}_i^{r,k} - \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k}) - \mu E_i g_i(\boldsymbol{w}_i^{r,k}) = \mathcal{T}_1 + \mathcal{T}_2 + \mathcal{T}_3,$$

where

$$\mathcal{T}_1 = \eta_i \Upsilon_i^{(r-1)K+k-1} \boldsymbol{w}_i^{1,0}, \quad (16)$$

$$
\begin{aligned}
\mathcal{T}_2 = -\eta_i \Upsilon_i \textstyle\sum_{j\in\mathcal{E}_i}&\{\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k} - \mu\eta_i E_i \Upsilon_i (\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k-1} + \\
&\Upsilon_i \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k-2} +, \ldots, + \Upsilon_i^{(r-1)K+k-1} \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{1,0})\}, \quad (17)
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{T}_3 = -\mu\eta_i E_i \Upsilon_i &\{g_i(\boldsymbol{w}_i^{r,k}) + \Upsilon_i g_i(\boldsymbol{w}_i^{r,k-1}) + \\
&\Upsilon_i^2 g_i(\boldsymbol{w}_i^{r,k-2}) +, \ldots, + \Upsilon_i^{(r-1)K+k} g_i(\boldsymbol{w}_i^{1,0})\}. \quad (18)
\end{aligned}
$$

When the number of update iterations is sufficient, $\mathcal{T}_1$ can be ignored because $\mathcal{T}_1 \rightarrow 0$. In $\mathcal{T}_2$ and $\mathcal{T}_3$, the summation of geometric progression weight is included. Investigating the infinite sum of this geometric progression, it is guaranteed to be one, independently of $\eta_i$-selection, as

$$\mu\eta_i E_i \Upsilon_i (1 + \Upsilon_i +, \ldots, + \Upsilon_i^{\infty}) = \tfrac{\mu\eta_i E_i \Upsilon_i}{1-\Upsilon_i} = 1. \quad (19)$$

This indicates that the weighted summation in (17) and (18) is an implementation of the expectation computation over both outer $(r)$ and inner iterations $(k)$ as

$$\mathcal{T}_2 = -\eta_i \Upsilon_i \textstyle\sum_{j\in\mathcal{E}_i} (\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k} - \mathbb{E}_{r,k}[\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}]), \quad (20)$$

$$\mathcal{T}_3 = -\mathbb{E}_{r,k}[g_i(\boldsymbol{w}_i)] = -\boldsymbol{c}_i, \quad (21)$$

where the mean of $\mathcal{T}_2$ will be zero and the time constant in the expectation computation is larger when increasing $\eta_i$. Since $\mathcal{T}_3$ corresponds to a local control variate as in (21), $\eta_i$ must be selected such that $\mathcal{T}_2$ is matched with the global control variate. From (11) and (13), we can set the optimal $\eta_i$ as $\eta_i \Upsilon_i = 1/(\mu K E_i)$, and this results in

$$\eta_i = 1/(\mu E_i (K-1)), \quad (22)$$

where the number of inner loops is imposed to be $K \geq 2$ since $\eta_i > 0$. Then, (15) is reformulated such that it follows the SVR update rule (3):

$$\boldsymbol{w}_i^{r,k+1} = \boldsymbol{w}_i^{r,k} - \mu\{g_i(\boldsymbol{w}_i^{r,k}) + \bar{\boldsymbol{c}}_i^{r,k} - \boldsymbol{c}_i^{r,k}\}, \quad (23)$$

where the gradient control variates are given by substituting (23) into (17) and (18) as

$$
\begin{aligned}
\bar{\boldsymbol{c}}_i^{r,k} &= -\tfrac{1}{\mu K E_i} \textstyle\sum_{j\in\mathcal{E}_i} \{\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k} - \tfrac{1}{K}(\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k-1} + \\
&(1-\tfrac{1}{K})\mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{r,k-2} +, \ldots, + (1-\tfrac{1}{K})^{(r-1)K+k-1} \mathbf{A}_{i|j}^{\mathrm{T}} \boldsymbol{z}_{i|j}^{1,0})\}, \quad (24)\\
\boldsymbol{c}_i^{r,k} &= \tfrac{1}{K}\{g_i(\boldsymbol{w}_i^{r,k}) + (1-\tfrac{1}{K}) g_i(\boldsymbol{w}_i^{r,k-1}) + \\
&(1-\tfrac{1}{K})^2 g_i(\boldsymbol{w}_i^{r,k-2}) +, \ldots, + (1-\tfrac{1}{K})^{(r-1)K+k} g_i(\boldsymbol{w}_i^{1,0})\}. \quad (25)
\end{aligned}
$$

We now have found the surprising fact that the gradient control variates of SVR are implicitly included in the primal-dual formalism (ECL) by optimally selecting $\eta_i$ as (22). That is, SVR originates from the model-constrained minimization problem in (4).

### 5.2. Update rules of ECL-ISVR

The final algorithm forms for our ECL-ISVR are summarized in Alg. 2, where $\boldsymbol{w}_i$-update procedure is obtained by substituting (22) into (15). Compared with ECL in Alg. 1, it is upgraded (i) to include the optimization of $\eta_i$, which results in the $\boldsymbol{w}_i$-update procedure matching SVR and (ii) to halve the communication requirement as only the lifted dual variable are transmitted between connected nodes (this communication requirement is the same as DSGD).

It remains an open question whether it is possible to recover the data from $\boldsymbol{y}_{j|i}$ since this variable reflects the statistical properties of the data. Since the lifted dual variable is basically an update difference of model variables as in (11) and (13), it will not be possible to estimate the data or even the model variable without tracking of $\boldsymbol{y}_{j|i}$ over a long, continuous time, with the initial model value $\boldsymbol{w}_i^{1,0}$.

## 6. Convergence analysis of ECL-ISVR

We conducted a convergence analysis for Alg. 2 for strongly convex, general convex, and non-convex cost functions. The detailed proofs are provided in the supplementary material. In our convergence analysis, we used the same mathematical techniques and proof strategies as the SCAFFOLD paper (Karimireddy et al., 2020).

**Proof sketch**: We model the variable variance due to asynchronous communication lags in the supplementary material. We combine these variances with the lemmas in the SCAFFOLD paper to complete our convergence analysis for Alg. 2. Note that difference between PDMM-ISVR and ADMM-ISVR is not considered because they just differs in model difference computation between connected nodes, as in (11) and (13). The final results are given in the Theorem 1:

**Theorem 1.** *Suppose that the functions* $\{f_i\}$ *satisfy (D1)* $\beta$*-Lipschitz smooth and (D3)* $\sigma^2$*-bounded variance. Then, in each of following cases with appropriate step-size setting, the output of ECL-ISVR (PDMM-ISVR and ADMM-ISVR) in Alg. 2 satisfies*

***Strongly convex**: $\{f_i\}$ satisfies (D2) $\alpha$-convex with $\alpha > 0$, $\mu \in [0, \min(\frac{1}{27\beta K}, \frac{1}{3\alpha K}))$, $R \geq \max(\frac{27\beta}{2\alpha}, \frac{3}{2})$, then*

$$\Omega_{cv} \leq \mathcal{O}\big(\tfrac{E_{\min}}{E_{\min}-1}\{\alpha D_0^2 \exp(-\min(\tfrac{\alpha}{27\beta K}, \tfrac{1}{3K})R) + \tfrac{\sigma^2}{\alpha RK}(3+\tfrac{12}{N})\}\big),$$

[3]*where $\Omega_{cv} = \frac{1}{N}\sum_{i\in\mathcal{N}} \mathbb{E}[(f_i(\boldsymbol{w}_i^R) - f_i(\boldsymbol{w}_i^*))]$, $D_0^2 = \frac{1}{N}\sum_{i\in\mathcal{N}}(\|f_i(\boldsymbol{w}_i^{1,0}) - f_i(\boldsymbol{w}_i^*)\|^2 + \|\nabla f_i(\boldsymbol{w}_i^*) - \mathbb{E}[\boldsymbol{c}_i^{1,0}]\|^2)$, and $E_{\min} = \min(E_i)$ denotes the minimum number of edges associated with a node and assumed to be $E_{\min} \geq 2$.*

***General convex**: $\{f_i\}$ satisfies (D2) $\alpha$-convex with $\alpha = 0$, $\mu \in [0, \frac{1}{27\beta K})$, $R \geq 1$, then*

$$\Omega_{cv} \leq \mathcal{O}\big(\tfrac{E_{\min}}{E_{\min}-1}\{\tfrac{\sigma D_0}{\sqrt{RKN}}\sqrt{3+\tfrac{12}{N}} + \tfrac{27\beta D_0^2}{R}\}\big).$$

***Non-convex**: $\mu \in [0, \frac{1}{24\beta K})$, $R \geq 1$, then*

$$\Omega_{nc} \leq \mathcal{O}\Big(\tfrac{3\sigma\sqrt{Q_0}}{2\sqrt{RKN}}\sqrt{1+\tfrac{18}{N}} + \tfrac{3\beta Q_0}{R}\}\Big),$$

*where $\Omega_{nc} = \frac{1}{N}\sum_{i\in\mathcal{N}} \mathbb{E}\|\nabla f_i(\boldsymbol{w}_i^{r,0})\|^2$ and $Q_0 = \frac{1}{N}\sum_{i\in\mathcal{N}}(f_i(\boldsymbol{w}_i^{1,0}) - f_i(\boldsymbol{w}^*))$.*

## 7. Numerical experiments

We evaluated the constructed algorithms by investigating the learning curves for the case that statistically heterogeneous data subsets are placed at the local nodes. We aim to identify algorithms that nearly reach the performance of the reference case where all data are available on a single node.

### 7.1. Experimental setup

**Data set/models**: We prepared the following three problem settings (T1)–(T3):

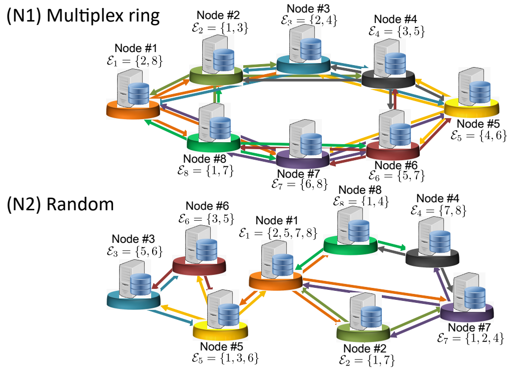
(N1) Multiplex ring
Node #1
$\mathcal{E}_1 = \{2,8\}$
Node #2
$\mathcal{E}_2 = \{1,3\}$
Node #3
$\mathcal{E}_3 = \{2,4\}$
Node #4
$\mathcal{E}_4 = \{3,5\}$
Node #5
$\mathcal{E}_5 = \{4,6\}$
Node #6
$\mathcal{E}_6 = \{5,7\}$
Node #7
$\mathcal{E}_7 = \{6,8\}$
Node #8
$\mathcal{E}_8 = \{1,7\}$
(N2) Random
Node #3
$\mathcal{E}_3 = \{5,6\}$
Node #6
$\mathcal{E}_6 = \{3,5\}$
Node #1
$\mathcal{E}_1 = \{2,5,7,8\}$
Node #8
$\mathcal{E}_8 = \{1,4\}$
Node #4
$\mathcal{E}_4 = \{7,8\}$
Node #7
$\mathcal{E}_7 = \{1,2,4\}$
Node #2
$\mathcal{E}_2 = \{1,7\}$
Node #5
$\mathcal{E}_5 = \{1,3,6\}$

*Figure 2.* Decentralized network composed of $N = 8$ local nodes with (N1) multiplex ring and (N2) random topologies

(T1) Fashion MNIST with a convex model: Fashion MNIST (Xiao et al., 2017) consists of $28 \times 28$ pixel of gray-scale images in 10 classes. The $60,000$ training samples are heterogeneously divided over $N = 8$ nodes. Each local node holds a different number of data and they are composed of 8 randomly selected classes out of a total of 10 classes. The same $10,000$ test images are used on all nodes for evaluation. For the classification task, we use logistic regression with an affine transformation. The associated logistic loss function is convex.

(T2) Fashion MNIST with a non-convex model: The data setting is the same as (T1). We applied a (non-convex) ResNet-32 (He et al., 2016) with minor modifications. Since the statistics in mini-batch samples are biased in our heterogeneous data setting, group normalization (Wu & He, 2018) (1 group per 32 ch) is used instead of batch normalization (Ioffe & Szegedy, 2015) before each convolutional layer. As a cost function, cross-entropy (CE) is used. $\boldsymbol{w}_i$ is initialized by He's method (He et al., 2015) with a common random seed.

(T3) CIFAR-10 with a non-convex model: The CIFAR-10 data set consists of $32\times32$ color images in 10 object classes (Krizhevsky et al., 2009), where $50,000$ training images are heterogeneously divided into $N = 8$ nodes. Each local node holds 8 randomly selected data classes. For evaluation, the same $10,000$ test images are used for all nodes. CE with ResNet-32 using group normalization is used as a non-convex cost function.

For all problem settings (T1)–(T3), squired $L_2$ model normalization with weight 0.01 is added to the cost function. The step-size $\mu = 0.002$ and the mini-batch size 100 are used in all settings. Since the mini-batch samples are uniformly selected from the unbalanced data subsets (8 out of 10 classes), gradient drift will occur in all settings.

**Networks**: To investigate the robustness in practical scenarios (heterogeneous data, asynchronous/sparse communication, and various network configurations), we prepared two network settings composed of $N = 8$ nodes, as in Fig. 2.

(N1) Multiplex ring topology: Each node is connected with its two neighboring nodes only ($E_i = 4$, $E = 32$).

[3] Further investigation is needed to provide a lower bound of $\Omega_{\text{cv}}$ for finite $R$. In principle, the model variables $\{\boldsymbol{w}_i^R\}$ tend to be the same as the iteration $R$ goes to $\infty$ due to the introduced control variants, which will eventually lead to nonnegativity of $\Omega_{\text{cv}}$.

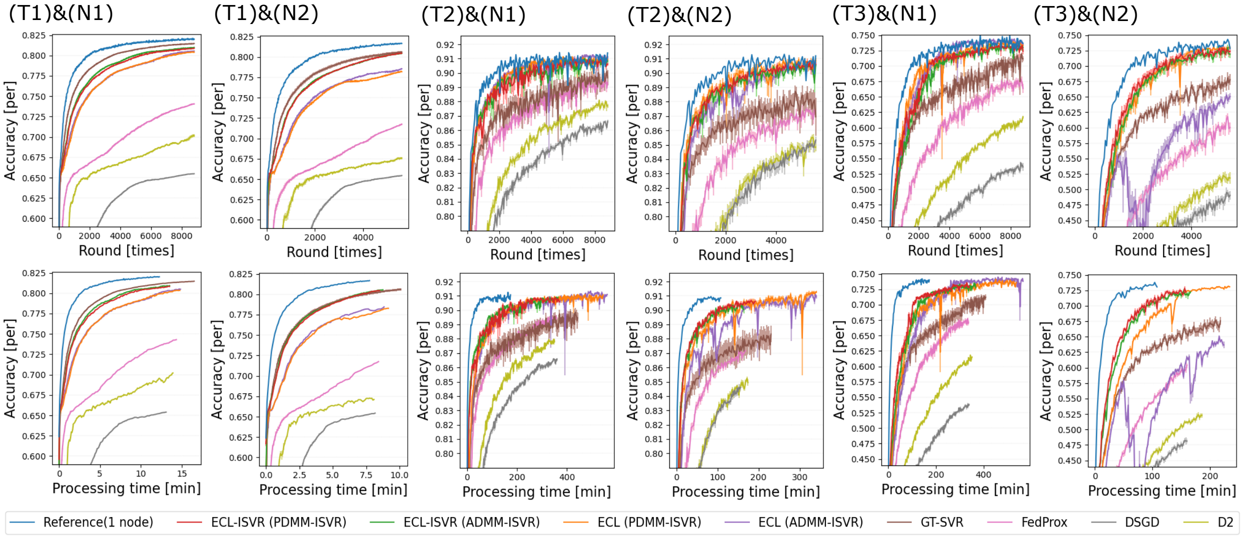

*Figure 3.* Learning curves using test data set of (T1)-(T3) for (N1) multiplex ring and (N2) random topologies of Fig. 2. Node-averaged classification accuracy is drawn as a solid line and one standard deviation is indicated by a light color.

(N2) Random topology: The number of edges for each local node is non-homogeneous ($E_i \in (2,4), E=20$) and identical over all experiments.

The communication between nodes was performed in an asynchronous manner as shown in Fig. 1. The communication for each edge is conducted once per $K=8$ local updates on average, with $R=8,800$ rounds for (N1) and $R=5,600$ rounds for (N2).

**Optimization algorithms**: The proposed ECL-ISVR (PDMM-ISVR and ADMM-ISVR) in Alg. 2 is compared with conventional algorithms, namely (1) DSGD, (2) FedProx, (3) GT-SVR (implemented by GT-SAGA (Xin et al., 2020)), (4) D$^2$ (Tang et al., 2018), and (5) ECL (PDMM-SGD and ADMM-SGD) in Alg. 1 with $\eta_i=3.0/\sqrt{e}, \rho_i=0.1$. In addition, we prepared a reference model, which is a global optimization model trained by applying vanilla SGD on a single node that has access to all data. The aim of all algorithms is to reach the score of the reference model.

**Implementation**: We constructed software that runs on a server that has 8 GPUs (NVIDIA GeForce RTX 2080Ti) with 2 CPUs (Intel Xeon Gold 5222, 3.80 GHz). PyTorch (v1.6.0) with CUDA (v10.2) and Gloo[4] for node communication was used. A part of our source code[5] is available.

### 7.2. Experimental results

Fig. 3 shows node-averaged classification accuracy learning curves for the test data set/model (T1)–(T3) for both the multiplex ring (N1) and random (N2) topologies. The horizontal axis displays the number of rounds in the upper row and the processing time in the lower row. While the processing time is the addition of (i) local node computation time for training/test data sets and (ii) communication time, these are shown separately in the supplementary material.

In almost all settings, PDMM-ISVR (ECL-ISVR) performed closest to the single-node reference scores and second-best was ADMM-ISVR (ECL-ISVR). The performance difference between them was slight. Their processing time was almost the same as that of DSGD, but it is slower than the single-node reference due to the heterogeneous data division over the $N$ nodes. Of the conventional algorithms, the performance with PDMM-SGD (ECL) was better, but its processing time was nearly 1.5 times that of DSGD. This is caused by the doubled communication requirement issue. In addition, ADMM-SGD (ECL) was unstable in some cases, as shown in (T2)&(N2) and (T3)&(N2). This may be caused by a non-optimal parameter choice $\{\eta_i, \rho_i\}$. GT-SVR takes the best score in (T1), however it did not reach the single-node reference scores in (T2) and (T3), even though the SVR technique was applied. This indicates that our control variate implementation (24)-(25) provides robust performance in practical scenarios. FedProx, D$^2$, and DSGD did not work well, likely because of gradient drift. The proposed ECL-ISVR appears to be robust to network configurations as the network configuration does not affect performance significantly. In summary, the experiments confirm the effectiveness of the proposed ECL-ISVR.

## 8. Conclusion

We succeeded in including the gradient control rules of SVR implicitly in the primal-dual formalism (ECL) by optimally selecting $\eta_i$ as (22). Through convergence analysis and experiments using convex/non-convex models, it was confirmed that the proposed ECL-ISVR algorithms work well in practical scenarios (heterogeneous data, asynchronous communication, arbitrary network configurations).

## Acknowledgements

We thank Masahiko Hashimoto and Tomonori Tada for their help with the experiments.

[4] https://pytorch.org/docs/stable/distributed.html

[5] https://github.com/nttcslab/ecl-isvr

## References

Bauschke, H. H., Combettes, P. L., et al. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer, 2011.

Blot, M., Picard, D., Cord, M., and Thome, N. Gossip training for deep learning. *arXiv preprint arXiv:1611.09726*, 2016.

Chen, J. and Sayed, A. H. Diffusion adaptation strategies for distributed optimization and learning over networks. *IEEE Transactions on Signal Processing*, 60(8):4289–4305, 2012.

Custers, B., Sears, A. M., Dechesne, F., Georgieva, I., Tani, T., and Van der Hof, S. *EU personal data protection in policy and practice*. Springer, 2019.

Defazio, A., Bach, F., and Lacoste-Julien, S. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1646–1654, 2014.

Douglas, J. and Rachford, H. H. On the numerical solution of heat conduction problems in two and three space variables. *Transactions of the American mathematical Society*, 82(2):421–439, 1956.

Fenchel, W. On conjugate convex functions. *Canadian Journal of Mathematics*, 1(1):73–77, 1949.

Gabay, D. and Mercier, B. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & mathematics with applications*, 2(1):17–40, 1976.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Jiang, Z., Balu, A., Hegde, C., and Sarkar, S. Collaborative deep learning in fixed topology networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5904–5914, 2017.

Jin, P. H., Yuan, Q., Iandola, F., and Keutzer, K. How to scale distributed deep learning? *arXiv preprint arXiv:1611.04581*, 2016.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 315–323, 2013.

Kar, S. and Moura, J. M. Consensus+ innovations distributed inference over networks: cooperation and sensing in networked systems. *IEEE Signal Processing Magazine*, 30(3):99–109, 2013.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. *1st Adaptive & Multitask Learning Workshop*, 2019.

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5330–5340, 2017.

Lim, W. Y. B., Luong, N. C., Hoang, D. T., Jiao, Y., Liang, Y.-C., Yang, Q., Niyato, D., and Miao, C. Federated learning in mobile edge networks: a comprehensive survey. *IEEE Communications Surveys & Tutorials*, 2020.

Mao, Y., You, C., Zhang, J., Huang, K., and Letaief, K. B. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, 19(4):2322–2358, 2017.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication–efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR, 2017.

Nesterov, Y. et al. *Lectures on convex optimization*, volume 137. Springer, 2018.

Niwa, K., Harada, N., Zhang, G., and Kleijn, W. B. Edge-consensus learning: Deep learning on p2p networks with nonhomogeneous data. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 668–678, 2020.

Ormándi, R., Hegedüs, I., and Jelasity, M. Gossip learning with linear models on fully distributed data. *Concurrency and Computation: Practice and Experience*, 25(4):556–571, 2013.

Pathak, R. and Wainwright, M. J. Fedsplit: An algorithmic framework for fast federated optimization. *34th Conference on Neural Inforamation Processing Systems (NeurIPS 2020)*, 2020.

Peaceman, D. W. and Rachford, Jr, H. H. The numerical solution of parabolic and elliptic differential equations. *Journal of the Society for industrial and Applied Mathematics*, 3(1):28–41, 1955.

Rajawat, K. and Kumar, C. A primal-dual framework for decentralized stochastic optimization. *arXiv preprint arXiv:2012.04402*, 2020.

Ram, S. S., Nedić, A., and Veeravalli, V. V. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of optimization theory and applications*, 147(3):516–545, 2010.

Rockafellar, R. T. *Convex analysis*. Number 28. Princeton university press, 1970.

Ryu, E. K. and Boyd, S. Primer on monotone operator methods. *Appl. Comput. Math*, 15(1):3–43, 2016.

Sattler, F., Wiedemann, S., Müller, K.-R., and Samek, W. Robust and communication-efficient federated learning from non-iid data. *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

Sherson, T. W., Heusdens, R., and Kleijn, W. B. Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory. *IEEE Transactions on Signal and Information Processing over Networks*, 5 (2):334–347, 2018.

Shi, W., Cao, J., Zhang, Q., Li, Y., and Xu, L. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

Tang, H., Lian, X., Yan, M., Zhang, C., and Liu, J. D$^2$: decentralized training over decentralized data. *arXiv preprint arXiv:1803.07068*, 2018.

Wu, Y. and He, K. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 3–19, 2018.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Xie, C., Koyejo, S., and Gupta, I. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*, 2019.

Xin, R., Khan, U. A., and Kar, S. Variance-reduced decentralized stochastic optimization with accelerated convergence. *IEEE Transactions on Signal Processing*, 2020.

Zhang, G. and Heusdens, R. Distributed optimization using the primal-dual methomcd of multipliers. *IEEE Transactions on Signal and Information Processing over Networks*, 4(1):173–187, 2017.

Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., and Zhang, J. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, 107(8):1738–1762, 2019.