Multi-Agent Optimization and Learning
Lecture 5: Performance Guarantees and Trade-Offs

Stefan Vlaski[†] and Ali H. Sayed[⋆]

[†]Department of Electrical and Electronic Engineering, Imperial College London, UK
[⋆]Adaptive Systems Laboratory, École Polytechnique Fédérale de Lausanne, Switzerland

IEEE ICASSP 2024 Short Course

Lecture Aims

- In the last lecture we derived a range of decentralized learning algorithms, which we classified as penalty-based, primal-dual and gradient-tracking based.
- We also derived incremental variants.
- In this lecture, we will develop convergence guarantees for all of these algorithms, which clarify the impact of:
  - Bias-correction
  - The step-size and gradient-noise
  - Network connectivity
  - Heterogeneity of local objectives

Network Basis Transformation

When we studied the dynamics of decentralized averaging algorithms in Lecture 3, we found it useful to separately study the evolution of the network centroid and that of each individual agent's deviation from the centroid. In the context of decentralized optimization algorithms, we will employ the same kind of technique. We illustrate this in the context of the distributed gradient descent algorithm, repeated here for reference in network quantities:

$$\mathcal{W}_i = \mathcal{A}^{\mathsf{T}}\, \mathcal{W}_{i-1} - \mu \nabla \mathcal{J}(\mathcal{W}_{i-1}) \tag{1}$$

or in terms of node quantities:

$$w_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} w_{\ell,i-1} - \mu \nabla J_k(w_{k,i-1}) \tag{2}$$

For generality, we will allow for stochastic gradient approximations, resulting in:

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k}\, \boldsymbol{w}_{\ell,i-1} - \mu \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{3}$$

where we replaced the true gradient $\nabla J_k(w_{k,i-1})$ by its stochastic approximation $\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1})$, and changed the iterates $\boldsymbol{w}_{k,i}$ to utilize bold font since they are now random.

Network Basis Transformation

In network notation, we can then write:

$$\boldsymbol{\mathcal{W}}_i = \mathcal{A}^{\mathsf{T}}\, \boldsymbol{\mathcal{W}}_{i-1} - \mu \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \tag{4}$$

where we defined the network gradient approximation:

$$\widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) = \begin{pmatrix} \widehat{\nabla J}_1(\boldsymbol{w}_{1,i-1}) \\ \widehat{\nabla J}_2(\boldsymbol{w}_{2,i-1}) \\ \vdots \\ \widehat{\nabla J}_K(\boldsymbol{w}_{K,i-1}) \end{pmatrix} \tag{5}$$

Network Basis Transformation

Now recall from Lecture 3, that the weight matrix $\mathcal{A}$, when generated from a strongly connected graph, is primitive, and as a result has a very structured Jordan decomposition decomposition $A = V_\epsilon J V_\epsilon^{-1}$:

$$V_\epsilon = \begin{bmatrix} p & V_R \end{bmatrix}, \quad J = \begin{bmatrix} 1 & 0 \\ 0 & J_\epsilon \end{bmatrix}, \quad V_\epsilon^{-1} = \begin{bmatrix} \mathbb{1}^\mathsf{T} \\ V_L^\mathsf{T} \end{bmatrix} \tag{6}$$

In this lecture, we will be employing symmetric combination matrices $A = A^\mathsf{T}$, in which case the Jordan decomposition reduces to the eigendecomposition, and we can more simply write $A = V \Lambda V^\mathsf{T}$ with $V^\mathsf{T} V = V V^\mathsf{T} = I_K$ and:

$$V = \begin{bmatrix} \frac{1}{\sqrt{K}}\mathbb{1} & V_2 \end{bmatrix}, \quad \Lambda = \begin{bmatrix} 1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} \tag{7}$$

The matrix Λ_2, which corresponds to the Jordan matrix J_ϵ, in (7) is now a diagonal matrix, with $\lambda_2(A)$ through $\lambda_K(A)$ on the diagonal and $\rho(\Lambda_2) < 1$.

Network Basis Transformation

Given the eigendecomposition of A, we can deduce the eigendecomposition of $\mathcal{A} = A \otimes I_M$ through the observation that:

$$A = V\Lambda V^{\mathsf{T}} \Longleftrightarrow \mathcal{A} = (V \otimes I_M)(\Lambda \otimes I_M)(V \otimes I_M)^{\mathsf{T}} = \mathcal{V}\Lambda\mathcal{V}^{\mathsf{T}} \tag{8}$$

where we defined:

$$\mathcal{V} = V \otimes I_M \tag{9}$$

$$\Lambda = \Lambda \otimes I_M \tag{10}$$

Network Basis Transformation

We use $\mathcal{V}$ to define a basis transformation for the distributed gradient descent recursion (4):

$$\begin{aligned}
\mathcal{V}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_i &= \mathcal{V}^{\mathsf{T}}\mathcal{A}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{i-1} - \mu\mathcal{V}^{\mathsf{T}}\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \\
&\overset{(a)}{=} \mathcal{V}^{\mathsf{T}}\mathcal{A}^{\mathsf{T}}\mathcal{V}\mathcal{V}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{i-1} - \mu\mathcal{V}^{\mathsf{T}}\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \\
&\overset{(b)}{=} \mathcal{V}^{\mathsf{T}}\mathcal{A}\mathcal{V}\mathcal{V}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{i-1} - \mu\mathcal{V}^{\mathsf{T}}\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \\
&\overset{(c)}{=} \Lambda\mathcal{V}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{i-1} - \mu\mathcal{V}^{\mathsf{T}}\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1})
\end{aligned} \tag{11}$$

where (a) follows since $\mathcal{V}$ is orthogonal, (b) follows by symmetry of A and (c) employs the eigendecomposition of A. If we define $\boldsymbol{\mathcal{W}}'_i \triangleq \mathcal{V}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_i$, we can write:

$$\boldsymbol{\mathcal{W}}'_i = \Lambda\boldsymbol{\mathcal{W}}'_{i-1} - \mu\mathcal{V}^{\mathsf{T}}\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \tag{12}$$

Network Basis Transformation

We find that the network basis transformation partially diagonalizes the network recursion, since $\Lambda = \Lambda \otimes I_M$ is block-diagonal. We say "partially" here because (12) is still driven by the term $\mu \mathcal{V}^{\mathsf{T}} \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{w}}_{i-1})$. We have:

$$\begin{aligned} \mathcal{V}^{\mathsf{T}} \boldsymbol{\mathcal{w}}_i = \left(\begin{bmatrix} \frac{1}{\sqrt{K}} \mathbb{1}^{\mathsf{T}} \\ V_2^{\mathsf{T}} \end{bmatrix} \otimes I_M \right) \boldsymbol{\mathcal{w}}_i &= \begin{bmatrix} \frac{1}{\sqrt{K}} \mathbb{1}^{\mathsf{T}} \otimes I_M \\ V_2^{\mathsf{T}} \otimes I_M \end{bmatrix} \boldsymbol{\mathcal{w}}_i \\ &= \begin{bmatrix} \frac{1}{\sqrt{K}} \sum_{k=1}^{K} \boldsymbol{w}_{k,i} \\ \mathcal{V}_2^{\mathsf{T}} \boldsymbol{\mathcal{w}}_i \end{bmatrix} \\ &\overset{(a)}{=} \begin{bmatrix} \sqrt{K}\, \boldsymbol{w}_{c,i} \\ \mathcal{V}_2^{\mathsf{T}} \boldsymbol{\mathcal{w}}_i \end{bmatrix} \end{aligned} \tag{13}$$

where in (a) we defined the network centroid $\boldsymbol{w}_{c,i} = \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{w}_{k,i}$. Hence, the first block captures the dynamics of the *network centroid*. The second block $\mathcal{V}_2^{\mathsf{T}} \boldsymbol{\mathcal{w}}_i$ also carries a useful interpretation, as we see on the next slide.

Deviation from the Centroid

Note that:

$$\boldsymbol{w}_i' = \mathcal{V}^{\mathsf{T}}\,\boldsymbol{w}_i \Longrightarrow \mathcal{V}\,\boldsymbol{w}_i' = \mathcal{V}\mathcal{V}^{\mathsf{T}}\,\boldsymbol{w}_i = \boldsymbol{w}_i \tag{14}$$

and hence

$$\begin{aligned}
\boldsymbol{w}_i &= \begin{bmatrix} \frac{1}{\sqrt{K}}\mathbb{1}\otimes I_M & \mathcal{V}_2 \end{bmatrix} \boldsymbol{w}_i' \\
&= \begin{bmatrix} \frac{1}{\sqrt{K}}\mathbb{1}\otimes I_M & \mathcal{V}_2 \end{bmatrix} \begin{bmatrix} \sqrt{K}\,\boldsymbol{w}_{c,i} \\ \mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{w}_i \end{bmatrix} \\
&= (\mathbb{1}\otimes I_M)\,\boldsymbol{w}_{c,i} + \mathcal{V}_2\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{w}_i \\
&= \mathbb{1}\otimes \boldsymbol{w}_{c,i} + \mathcal{V}_2\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{w}_i
\end{aligned} \tag{15}$$

After rearranging, we have:

$$\boldsymbol{w}_i - \mathbb{1}\otimes \boldsymbol{w}_{c,i} = \mathcal{V}_2\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{w}_i \tag{16}$$

We see that $\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{w}_i$ captures information about the deviation of each agent $\boldsymbol{w}_{k,i}$ from the centroid $\boldsymbol{w}_{c,i}$.

Network Basis Transformation

For the right-hand side, we have:

$$
\begin{aligned}
\begin{bmatrix} \sqrt{K}\,\boldsymbol{w}_{c,i} \\ \mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i \end{bmatrix} &= \begin{bmatrix} I_M & 0 \\ 0 & \Lambda_2 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{K}}\mathbb{1}^{\mathsf{T}} \otimes I_M \\ \mathcal{V}_2^{\mathsf{T}} \end{bmatrix} \boldsymbol{\mathcal{W}}_{i-1} - \mu \begin{bmatrix} \frac{1}{\sqrt{K}}\mathbb{1}^{\mathsf{T}} \otimes I_M \\ \mathcal{V}_2^{\mathsf{T}} \end{bmatrix} \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \\
&= \begin{bmatrix} \frac{1}{\sqrt{K}}\mathbb{1}^{\mathsf{T}} \otimes I_M \\ \Lambda_2 \mathcal{V}_2^{\mathsf{T}} \end{bmatrix} \boldsymbol{\mathcal{W}}_{i-1} - \mu \begin{bmatrix} \frac{1}{\sqrt{K}}\mathbb{1}^{\mathsf{T}} \otimes I_M \\ \mathcal{V}_2^{\mathsf{T}} \end{bmatrix} \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \\
&= \begin{bmatrix} \left(\frac{1}{\sqrt{K}}\mathbb{1}^{\mathsf{T}} \otimes I_M\right) \boldsymbol{\mathcal{W}}_{i-1} \\ \Lambda_2 \mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_{i-1} \end{bmatrix} - \mu \begin{bmatrix} \left(\frac{1}{\sqrt{K}}\mathbb{1}^{\mathsf{T}} \otimes I_M\right) \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \\ \mathcal{V}_2^{\mathsf{T}} \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \end{bmatrix} \\
&= \begin{bmatrix} \frac{1}{\sqrt{K}} \sum_{k=1}^{K} \boldsymbol{w}_{k,i-1} \\ \Lambda_2 \mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_{i-1} \end{bmatrix} - \mu \begin{bmatrix} \frac{1}{\sqrt{K}} \sum_{k=1}^{K} \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \\ \mathcal{V}_2^{\mathsf{T}} \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \end{bmatrix} \\
&= \begin{bmatrix} \sqrt{K}\,\boldsymbol{w}_{c,i-1} \\ \Lambda_2 \mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_{i-1} \end{bmatrix} - \mu \begin{bmatrix} \frac{1}{\sqrt{K}} \sum_{k=1}^{K} \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \\ \mathcal{V}_2^{\mathsf{T}} \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \end{bmatrix}
\end{aligned}
\tag{17}
$$

Partially Decoupled Recursions

We observe that the transformed recursion partially decouples into two recursions:

$$\sqrt{K}\,\boldsymbol{w}_{c,i} = \sqrt{K}\,\boldsymbol{w}_{c,i-1} - \frac{\mu}{\sqrt{K}} \sum_{k=1}^{K} \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{18}$$

$$\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i = \Lambda_2 \mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_{i-1} - \mu \mathcal{V}_2^{\mathsf{T}} \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \tag{19}$$

We divide (18) by $\sqrt{K}$ and find:

$$\boldsymbol{w}_{c,i} = \boldsymbol{w}_{c,i-1} - \frac{\mu}{K} \sum_{k=1}^{K} \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{20}$$

$$\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i = \Lambda_2 \mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_{i-1} - \mu \mathcal{V}_2^{\mathsf{T}} \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \tag{21}$$

Examination of (20) reveals that the network centroid $\boldsymbol{w}_{c,i}$ evolves *almost* as the centralized stochastic gradient algorithm we studied in Lecture 1, except that stochastic gradient approximations $\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1})$ are evaluated at the local iterates $\boldsymbol{w}_{k,i-1}$ rather than the centroid $\boldsymbol{w}_{c,i-1}$.

Network Basis Transformation for Diffusion

Recall that we motivated in Lecture 4 the Adapt-then-Combine (ATC) diffusion algorithm by employing an incremental variation of the argument that led to the distributed gradient descent algorithm:

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{22}$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \tag{23}$$

We are again allowing for stochastic gradient approximations $\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1})$. In terms of network quantities, we have:

$$\boldsymbol{\mathcal{W}}_i = \mathcal{A}^{\mathsf{T}} \left(\boldsymbol{\mathcal{W}}_{i-1} - \mu \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \right) \tag{24}$$

Network Decomposition for Diffusion

Applying the same network basis transformation $\boldsymbol{\mathcal{W}}_i' = \mathcal{V}^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i$, and repeating the argument that led to the decomposition (20)–(21), we find for the diffusion algorithm:

$$\boldsymbol{w}_{c,i} = \boldsymbol{w}_{c,i-1} - \frac{\mu}{K}\sum_{k=1}^{K}\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{25}$$

$$\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i = \Lambda_2\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_{i-1} - \mu\Lambda_2\mathcal{V}_2^{\mathsf{T}}\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \tag{26}$$

Note that the recursions for the network centroid (25) and (20) are identical. Second, the recursions for the network deviation (26) and (21) are structurally similar, but distinguished by an additional factor Λ_2 multiplying the driving term $-\mu\Lambda_2\mathcal{V}_2^{\mathsf{T}}\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1})$ in the case of the diffusion algorithm. This factor results from the fact that the mixing operation in the case of the diffusion algorithm is applied to both the weights themselves as well as the gradient update. This subtle difference is the source of improved stability properties of incremental-type algorithms.

Network Decomposition for Penalty-Based Algorithms

We remark that the fact that the centroid recursions for distributed gradient descent and diffusion algorithms are identical is a useful insight, but does *not* imply that the trajectories of both centroids will be identical. This is because the centroid recursions are coupled with the deviation recursions (26) and (21) through the iterates $\boldsymbol{w}_{k,i-1}$, where the gradient approximations are evaluated. These distinctions will seep into the centroid recursions and cause varying dynamics of the centroid as well. Nevertheless, the structural similarity of the recursion allows us to formulate a general form of penalty-based algorithms, and develop convergence analysis that will apply to both algorithms. In particular, we will study a decomposition of the form:

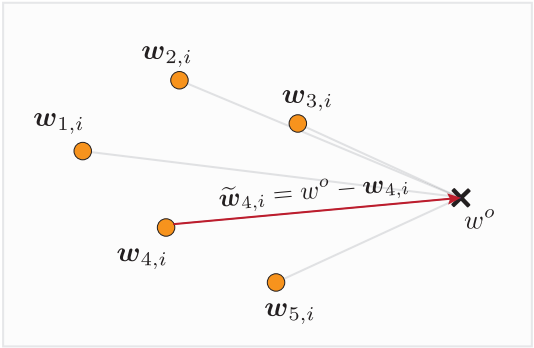
$$\boldsymbol{w}_{c,i} = \boldsymbol{w}_{c,i-1} - \frac{\mu}{K}\sum_{k=1}^{K}\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{27}$$

$$\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i = \Lambda_2\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_{i-1} - \mu\mathcal{D}\mathcal{V}_2^{\mathsf{T}}\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) \tag{28}$$

where we recover the distributed gradient descent algorithm by setting $\mathcal{D} = I$, and the diffusion algorithms by setting $\mathcal{D} = \Lambda_2$.

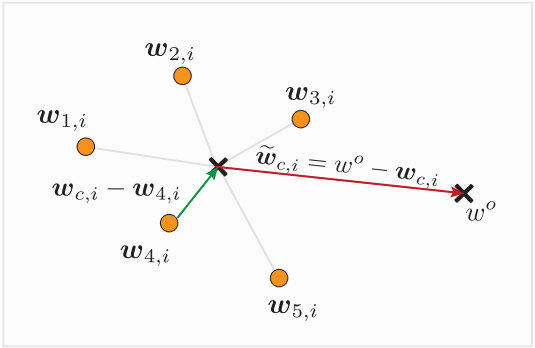
Visualization of the Network Decomposition

Individual error dynamics



Strong coupling of optimization and network effects through the total individual errors

Network error decomposition



Weak coupling of optimization effect, which affects primarily the centroid, and the network effect, which affects primarily the local deviations from the centroid

Perturbation Terms

To make the coupling explicit, we introduce the error terms:

$$\boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1}) \triangleq \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) - \nabla J_k(\boldsymbol{w}_{k,i-1}) \tag{29}$$

$$\boldsymbol{d}_{k,i-1}(\boldsymbol{w}_{k,i-1}) \triangleq \nabla J_k(\boldsymbol{w}_{k,i-1}) - \nabla J_k(\boldsymbol{w}_{c,i-1}) \tag{30}$$

We can then write:

$$\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) = \nabla J_k(\boldsymbol{w}_{c,i-1}) + \boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1}) + \boldsymbol{d}_{k,i-1}(\boldsymbol{w}_{k,i-1}) \tag{31}$$

Here, $\boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1})$ corresponds to a *gradient noise* term analogous to the ones we have encountered in previous non-cooperative, centralized and federated implementations of stochastic gradient algorithms. The second term $\boldsymbol{d}_{k,i-1}(\boldsymbol{w}_{k,i-1})$ represents a second deviation term resulting from lack of consensus across the network. In particular, as long as $\boldsymbol{w}_{k,i-1} \approx \boldsymbol{w}_{c,i-1}$, we would expect $\boldsymbol{d}_{k,i-1}(\boldsymbol{w}_{k,i-1})$ to be small. We define the network versions of these quantities as well:

$$\boldsymbol{s}_i(\boldsymbol{\mathcal{W}}_{i-1}) = \operatorname{col}\{\boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1})\} \tag{32}$$

$$\boldsymbol{d}_{i-1}(\boldsymbol{\mathcal{W}}_{i-1}) = \operatorname{col}\{\boldsymbol{d}_{k,i-1}(\boldsymbol{w}_{k,i-1})\} \tag{33}$$

Perturbed Transformed Recursions

These definitions allow us to reformulate the recursions (27)–(28) as:

$$\begin{aligned}\boldsymbol{w}_{c,i} &= \boldsymbol{w}_{c,i-1} - \frac{\mu}{K}\sum_{k=1}^{K}\nabla J_k(\boldsymbol{w}_{c,i-1}) - \frac{\mu}{K}\sum_{k=1}^{K}\left(\boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1}) + \boldsymbol{d}_{k,i-1}(\boldsymbol{w}_{k,i-1})\right) \\ &= \boldsymbol{w}_{c,i-1} - \mu\nabla J(\boldsymbol{w}_{c,i-1}) - \frac{\mu}{K}\sum_{k=1}^{K}\left(\boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1}) + \boldsymbol{d}_{k,i-1}(\boldsymbol{w}_{k,i-1})\right) \qquad (34)\end{aligned}$$

$$\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i = \Lambda_2\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_{i-1} - \mu\mathcal{D}\mathcal{V}_2^{\mathsf{T}}\nabla\mathcal{J}(\mathbb{1}\otimes\boldsymbol{w}_{c,i-1}) - \mu\mathcal{D}\mathcal{V}_2^{\mathsf{T}}\left(\boldsymbol{s}_i(\boldsymbol{\mathcal{W}}_{i-1}) + \boldsymbol{d}_{i-1}(\boldsymbol{\mathcal{W}}_{i-1})\right) \qquad (35)$$

The first recursion contracts in the mean-square sense for sufficiently small step-sizes since $\boldsymbol{w}_{c,i-1} - \frac{\mu}{K}\sum_{k=1}^{K}\nabla J_k(\boldsymbol{w}_{c,i-1})$ corresponds to a gradient step. The second recursion contracts since for strongly-connected graphs we have $\rho(\Lambda_2) < 1$.

Modeling Conditions

As we have done so far, we assume $J(w)$ to be ν-strongly convex with δ-Lipschitz gradients, and that the individual objectives have δ_k-Lipschitz gradients. Furthermore, we impose the gradient noise condition:

$$\mathbb{E}\left\{\boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1})\middle|\,\boldsymbol{w}_{k,i-1}\right\} = 0 \tag{36}$$

$$\mathbb{E}\left\{\|\boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1})\|^2\middle|\,\boldsymbol{w}_{k,i-1}\right\} \leq \beta_k^2\|w_k^o - \boldsymbol{w}_{k,i-1}\|^2 + \sigma_k^2 \tag{37}$$

Coupled Inequality Recursion

Upon bounding the perturbation terms we find one coupled inequality recursion:

$$\begin{bmatrix} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{c,i}\|^2 \\ \mathbb{E}\|\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i\,\|^2 \end{bmatrix} \leq \Gamma \begin{bmatrix} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{c,i-1}\|^2 \\ \mathbb{E}\|\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_{i-1}\,\|^2 \end{bmatrix} + \begin{bmatrix} \mu^2\sigma^2 \\ \mu^2 b \end{bmatrix} \tag{38}$$

where

$$\lambda = 1 - 2\mu\nu + \mu^2\delta^2 \tag{39}$$

$$\Gamma = \begin{bmatrix} \sqrt{\lambda} + O(\mu^2) & O\left(\frac{\mu}{K\nu}\right) \\ O\left(\frac{\mu^2\|\mathcal{D}\|^2}{1-\lambda_2}\right) & \lambda_2 + O(\mu^2) \end{bmatrix} \tag{40}$$

$$\sigma^2 = \frac{1}{K^2}\sum_{k=1}^{K}\left(3\beta_k^2\|w_k^o - w^o\|^2 + \sigma_k^2\right) \tag{41}$$

$$b = \frac{4}{1-\lambda_2}\|\mathcal{D}\|^2\|\mathcal{V}^{\mathsf{T}}\|^2\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2 + \|\mathcal{D}\|^2\|\mathcal{V}^{\mathsf{T}}\|^2K^2\sigma^2 \tag{42}$$

Mean-square Behavior

Theorem (Mean-square-behavior of penalty-based decentralized algorithms)

There exists a step-size μ that is small enough, so that:

$$\rho(\Gamma) \leq \|\Gamma\|_1 = \max\left\{1 - \mu\nu + O(\mu^2), \lambda_2 + O(\mu)\right\} < 1 \tag{43}$$

and the decentralized penalty-based algorithm converge in the sense that:

$$\begin{bmatrix} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{c,i}\|^2 \\ \mathbb{E}\|\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i\,\|^2 \end{bmatrix} \leq O\left(\rho(\Gamma)^i\right) + \begin{bmatrix} O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 b}{K(1-\lambda_2)}\right) \\ O\left(\frac{\mu^2 b}{1-\lambda_2}\right) + O\left(\frac{\mu^3\sigma^2}{(1-\lambda_2)^2}\right) \end{bmatrix} \tag{44}$$

where

$$b = \frac{4}{1-\lambda_2}\|\mathcal{D}\|^2\|\mathcal{V}^{\mathsf{T}}\|^2\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2 + \|\mathcal{D}\|^2\|\mathcal{V}^{\mathsf{T}}\|^2K^2\sigma^2 \tag{45}$$

Returning to Individual Errors

The theorem quantifies separately the centroid error along with the deviation of individual agents from the centroid. We can return to individual errors $\widetilde{\boldsymbol{w}}_{k,i} = w^o - \boldsymbol{w}_{k,i}$ by noting:

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2 = \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|w^o - \boldsymbol{w}_{c,i} + \boldsymbol{w}_{c,i} - \boldsymbol{w}_{k,i}\|^2 \leq 2\mathbb{E}\|\widetilde{\boldsymbol{w}}_{c,i}\|^2 + \frac{2}{K}\sum_{k=1}^{K}\mathbb{E}\|\,\boldsymbol{w}_{c,i} - \boldsymbol{w}_{k,i}\|^2 \tag{46}$$

where the last step follows from Jensen's inequality. The first term corresponds to the centroid error, while the last term can be related to $\mathbb{E}\|\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i\,\|^2$ via:

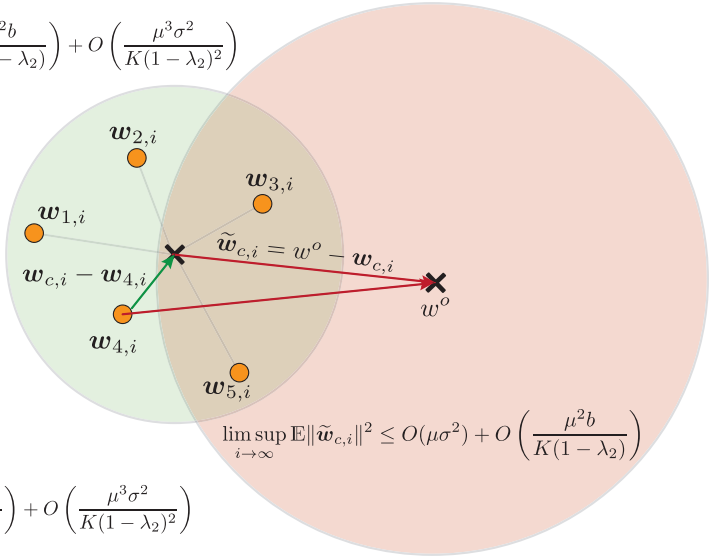
$$\frac{2}{K}\sum_{k=1}^{K}\mathbb{E}\|\,\boldsymbol{w}_{c,i} - \boldsymbol{w}_{k,i}\|^2 = \frac{2}{K}\mathbb{E}\|\,\boldsymbol{\mathcal{W}}_i - \mathbb{1}\otimes\boldsymbol{w}_{c,i}\,\|^2 = \frac{2}{K}\mathbb{E}\|\mathcal{V}_2\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i\,\|^2 \overset{(16)}{\leq} \frac{2\|\mathcal{V}_2\|^2}{K}\mathbb{E}\|\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i\,\|^2$$

Hence:

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2 \leq 2\mathbb{E}\|\widetilde{\boldsymbol{w}}_{c,i}\|^2 + \frac{2\|\mathcal{V}_2\|^2}{K}\mathbb{E}\|\mathcal{V}_2^{\mathsf{T}}\,\boldsymbol{\mathcal{W}}_i\,\|^2 \tag{47}$$

Visualization of the Limiting Regions

$$\limsup_{i\to\infty} \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}\|\boldsymbol{w}_{k,i} - \boldsymbol{w}_{c,i}\|^2 \leq O\left(\frac{\mu^2 b}{K(1-\lambda_2)}\right) + O\left(\frac{\mu^3\sigma^2}{K(1-\lambda_2)^2}\right)$$



$$\limsup_{i\to\infty} \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2 \leq O(\mu\sigma^2) + O\left(\frac{\mu^2 b}{K(1-\lambda_2)}\right) + O\left(\frac{\mu^3\sigma^2}{K(1-\lambda_2)^2}\right)$$

Discussion

Assuming the step-size is small enough so that terms of order 3 and higher can be disregarded, we find for the average mean-squared deviation:

$$\limsup_{i\to\infty} \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2 \leq O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 b}{K(1-\lambda_2)}\right) \tag{48}$$

Here, μ denotes the step-size of the algorithm and λ_2 denotes the second-largest eigenvalue of the weight matrix A. The remaining constants are:

$$\sigma^2 = \frac{1}{K^2}\sum_{k=1}^{K}\left(3\beta_k^2\|w_k^o - w^o\|^2 + \sigma_k^2\right) \tag{49}$$

$$b = O\left(\frac{\|\mathcal{D}\|^2\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2}{1-\lambda_2} + \|\mathcal{D}\|^2K^2\sigma^2\right) \tag{50}$$

Leading-Order Term and Linear Gain

The first term $O(\mu\sigma^2)$ is proportional to the step-size μ and the aggregate gradient noise σ^2. We encountered this exact term as the steady-state-error expression for stochastic gradient-descent in earlier lectures. Indeed, after specializing to the homogeneous scenario where all local gradient noise profiles $\sigma_k^2 = \sigma_1^2$ and minimizers $w_k^o = w^o$ are the same, we can recover linear performance gain via:

$$\sigma^2 = \frac{1}{K^2}\sum_{k=1}^{K}\left(3\beta_k^2\|w_k^o - w^o\|^2 + \sigma_k^2\right) = \frac{1}{K^2}\sum_{k=1}^{K}\sigma_k^2 = \frac{\sigma_1^2}{K} \tag{51}$$

and hence:

$$O(\mu\sigma^2) = O\left(\frac{\mu\sigma_1^2}{K}\right) \tag{52}$$

We conclude that in a homogeneous setting, and for small step-sizes, K agents will perform K times better than a single agent.

Higher-Order Term and the Cost of Decentralization

The second term $\frac{\mu^2 b}{K(1-\lambda_2)}$ is new and a result of our decentralized implementation. It essentially corresponds to the loss in performance we endure since we are implementing our decentralized algorithm over a graph and rely on the local diffusion of estimate rather than central aggregation. For the diffusion algorithm, we have $\mathcal{D} = \Lambda_2$, and hence

$$\frac{\mu^2 b}{K(1-\lambda_2)} = O\left(\frac{\mu^2 \lambda_2^2 \sum_{k=1}^{K} \|\nabla J_k(w^o)\|^2}{K(1-\lambda_2)^2} + \frac{\mu^2 \lambda_2^2 K \sigma^2}{1-\lambda_2}\right) \tag{53}$$

This entire term is multiplied by μ^2. Assuming all other constants are fixed and finite, this means that as $\mu \to 0$, the bias term $\frac{\mu^2 b}{K(1-\lambda_2)}$ will eventually be dominated by the noise term $\mu\sigma^2$. Similarly, if the network is very densely connected, this will imply that $\lambda_2 \to 0$, and again the bias term will be dominated by the noise term, since both expressions on the right-hand side of (53) are scaled by λ_2^2.

The Effect of Heterogeneity

There are, however, important settings where the bias is non-trivial.

- When the network is very sparsely connected, resulting in λ_2 close to one and hence $1-\lambda_2 \to 0$. The fact that the two terms on the right-hand side of (53) are divided by $(1-\lambda_2)^2$ and $1-\lambda_2$ respectively has the potential to significantly amplify the bias term for sparse networks.
- When exact gradients are used, or the gradient approximation is of very high quality, resulting in $\sigma^2 \to 0$. In that case the term $\frac{\mu^2\lambda_2^2 \sum_{k=1}^K \|\nabla J_k(w^o)\|^2}{(1-\lambda_2)^2}$ will dominate all terms involving the gradient noise variance σ^2 and cause a bottleneck.

This insight is consistent with the discussion in Lecture 4, which concluded that the distributed gradient descent algorithm is unbiased if, and only if, all objectives $J_k(w)$ are minimized at a common minimizer w^o, which implies $\sum_{k=1}^K \|\nabla J_k(w^o)\|^2 = 0$. We will hence now investigate if bias-corrected algorithms yield improved performance guarantees.

Unified Formulation

In this section, we describe a unifying and generalized framework that includes *all* the decentralized primal-dual and gradient tracking-based methods derived so far as special cases. Let $\mathcal{B} \in \mathbb{R}^{KM \times KM}$ and $\mathcal{C} \in \mathbb{R}^{KM \times KM}$ denote two general *symmetric* matrices that satisfy the following conditions:

$$\begin{cases} \mathcal{B}\,\mathcal{W} = 0 \iff w_1 = w_2 \ldots, w_K \\ \mathcal{C}\,\mathcal{W} = 0 \iff \mathcal{B}\,\mathcal{W} = 0 \text{ or } \mathcal{C} = 0 \\ \mathcal{C} \text{ is positive semi-definite} \end{cases} \tag{54}$$

For example, $\mathcal{C} = \mathcal{L} = I_{KM} - \mathcal{A}$ and $\mathcal{B} = \mathcal{L}^{1/2}$ is one choice, but many other choices are possible including beyond what we have encountered so far. We will provide more examples in the sequel. Let also

$$\bar{\mathcal{A}} = \bar{A} \times I_M \tag{55}$$

where $\bar{A}$ is some symmetric doubly-stochastic and positive definite matrix. For example, $\bar{A} = \frac{1}{2}(I_K + A)$ is one possibility.

Unified Decentralized Algorithm

Assuming the matrices $\{\bar{\mathcal{A}}, \mathcal{B}, \mathcal{C}\}$ have been chosen, we can then solve:

$$\mathcal{W}^\star \triangleq \underset{\mathcal{W}\in\mathbb{R}^{KM}}{\operatorname{argmin}} \left\{ \mathcal{J}(\mathcal{W}) + \frac{1}{2\mu}\|\mathcal{W}\|_{\mathcal{C}}^2 \right\}, \quad \text{subject to } \mathcal{B}\mathcal{W} = 0 \tag{56}$$

and introduce the corresponding saddle-point formulation

$$\min_{\mathcal{W}} \max_{\lambda} \; \mathcal{J}(\mathcal{W}) \; + \; \frac{1}{2\mu}\|\mathcal{W}\|_{\mathcal{C}}^2 \; + \; \frac{1}{\mu}\lambda^{\mathsf{T}}\mathcal{B}\mathcal{W} \tag{57}$$

where $\lambda \in \mathbb{R}^{KM}$ is a Lagrangian factor and $\mu > 0$. To solve the above problem, we introduce the following *unified decentralized algorithm* (UDA), which consists of three successive steps (primal-descent, dual-ascent, and combination):

$$\boldsymbol{\mathcal{Z}}_i = (I_{KM} - \mathcal{C})\,\boldsymbol{\mathcal{W}}_{i-1} - \mu\,\widehat{\nabla\,\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) - \mathcal{B}\boldsymbol{\lambda}_{i-1} \tag{58}$$

$$\boldsymbol{\lambda}_i = \boldsymbol{\lambda}_{i-1} + \mathcal{B}\,\boldsymbol{\mathcal{Z}}_i \tag{59}$$

$$\boldsymbol{\mathcal{W}}_i = \bar{\mathcal{A}}\,\boldsymbol{\mathcal{Z}}_i \tag{60}$$

Recovering Decentralized Algorithms as Special Cases of UDA

Table: Obtaining several decentralized methods as special cases of the unified decentralized algorithm (UDA) described by (58)–(60). We define $A = I - L$ in terms of the Laplacian matrix L.

Algorithm	$\bar{\mathcal{A}}$	$\mathcal{B}$	$\mathcal{C}$
EXTRA	I_{KM}	$\mathcal{L}^{1/2}$	$\mathcal{L} = I_{KM} - \mathcal{A}$
EXACT diffusion	$\mathcal{A} = I_{KM} - \mathcal{L}$	$\mathcal{L}^{1/2}$	0
DIGing	I_{KM}	$\mathcal{L} = I_{KM} - \mathcal{A}$	$I_{KM} - \mathcal{A}^2$
NEXT	$\mathcal{A}$	$\mathcal{L} = I_{KM} - \mathcal{A}$	$I_{KM} - \mathcal{A}$
Aug-DGM	$\mathcal{A}^2$	$\mathcal{L} = I_{KM} - \mathcal{A}$	0

Network Decomposition for Bias-Corrected Algorithms

The argument essentially mirrors the one one for penalty-based methods. We will again decompose the network recursion into a recursion for the network centroid, and a second coupled recursion for the network deviation. The additional technical challenge now will be to account for the presence and impact of the dual variable $\boldsymbol{\lambda}_i$, which will complicate the recursions but ultimately be instrumental for bias correction. We can combine (58) and (60)

$$\boldsymbol{\mathcal{W}}_i = \bar{\mathcal{A}}(I_{KM} - \mathcal{C})\,\boldsymbol{\mathcal{W}}_{i-1} - \mu\,\bar{\mathcal{A}}\widehat{\nabla\,\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) - \bar{\mathcal{A}}\mathcal{B}\boldsymbol{\lambda}_{i-1} \tag{61}$$

Note that for all choices of $\bar{\mathcal{A}}, \mathcal{B}, \mathcal{C}$ in Table 1, we have:

$$\left(\mathbb{1}^{\mathsf{T}} \otimes I_M\right)\bar{\mathcal{A}} = \mathbb{1}^{\mathsf{T}} \otimes I_M \tag{62}$$

$$\left(\mathbb{1}^{\mathsf{T}} \otimes I_M\right)\mathcal{B} = 0 \tag{63}$$

$$\left(\mathbb{1}^{\mathsf{T}} \otimes I_M\right)(I_{KM} - \mathcal{C}) = \mathbb{1}^{\mathsf{T}} \otimes I_M \tag{64}$$

Network Centroid for Bias-Corrected Algorithms

We can then conclude:

$$\begin{aligned}
&\left(\mathbb{1}^{\mathsf{T}} \otimes I_M\right) \boldsymbol{\mathcal{W}}_i \\
&= \left(\mathbb{1}^{\mathsf{T}} \otimes I_M\right) \bar{\mathcal{A}}(I_{KM} - \mathcal{C})\, \boldsymbol{\mathcal{W}}_{i-1} - \mu \left(\mathbb{1}^{\mathsf{T}} \otimes I_M\right) \bar{\mathcal{A}} \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) - \left(\mathbb{1}^{\mathsf{T}} \otimes I_M\right) \bar{\mathcal{A}} \mathcal{B} \boldsymbol{\lambda}_{i-1} \\
&= \left(\mathbb{1}^{\mathsf{T}} \otimes I_M\right) \boldsymbol{\mathcal{W}}_{i-1} - \mu \left(\mathbb{1}^{\mathsf{T}} \otimes I_M\right) \widehat{\nabla \mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1})
\end{aligned} \tag{65}$$

Hence:

$$\boldsymbol{w}_{c,i} \triangleq \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{w}_{k,i} = \boldsymbol{w}_{c,i-1} - \frac{\mu}{K} \sum_{k=1}^{K} \widehat{\nabla J_k}(\boldsymbol{w}_{k,i-1}) \tag{66}$$

We conclude that the network centroid for all primal-dual and gradient tracking-based algorithms evolve according to an approximate centralized stochastic gradient recursion, provided that the matrices $\bar{\mathcal{A}}, \mathcal{B}, \mathcal{C}$ satisfy conditions (62)–(64).

Deviation from the Centroid

To quantify the deviation from the centroid, note that:

$$\boldsymbol{\mathcal{W}}_i - \mathbb{1} \otimes \boldsymbol{w}_{c,i} = \boldsymbol{\mathcal{W}}_i - \left(\frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}} \otimes I_M\right)\boldsymbol{\mathcal{W}}_i = \left(I_{KM} - \frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}} \otimes I_M\right)\boldsymbol{\mathcal{W}}_i \tag{67}$$

Applying this linear transformation to (61):

$$\begin{aligned}
&\boldsymbol{\mathcal{W}}_i - \mathbb{1} \otimes \boldsymbol{w}_{c,i} \\
&\overset{(a)}{=} \left(\bar{\mathcal{A}} - \frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}} \otimes I_M\right)(I_{KM} - \mathcal{C})\,(\boldsymbol{\mathcal{W}}_{i-1} - \mathbb{1} \otimes \boldsymbol{w}_{c,i-1}) \\
&\quad - \mu\left(\bar{\mathcal{A}} - \frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}} \otimes I_M\right)\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) - \left(\bar{\mathcal{A}} - \frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}} \otimes I_M\right)\mathcal{B}\boldsymbol{\lambda}_{i-1} \\
&\overset{(b)}{=} \mathcal{D}(I_{KM} - \mathcal{C})\,(\boldsymbol{\mathcal{W}}_{i-1} - \mathbb{1} \otimes \boldsymbol{w}_{c,i-1}) - \mu\,\mathcal{D}\widehat{\nabla\mathcal{J}}(\boldsymbol{\mathcal{W}}_{i-1}) - \mathcal{D}\mathcal{B}\boldsymbol{\lambda}_{i-1}
\end{aligned} \tag{68}$$

where in (a) we again made use of the spectral properties (62)–(64) which imply that $\left(\bar{\mathcal{A}} - \frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}} \otimes I_M\right)(I_{KM} - \mathcal{C})\,(\mathbb{1} \otimes \boldsymbol{w}_{c,i-1}) = 0$, and in (b) we defined:

$$\mathcal{D} = \bar{\mathcal{A}} - \frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}} \otimes I_M \tag{69}$$

Coupled Inequality Recursions

This time, we measure the deviation from the network centroid via:

$$\Delta_i \triangleq \mathbb{E}\|\boldsymbol{\mathcal{W}}_i - \mathbb{1}\otimes \boldsymbol{w}_{c,i}\|^2_{I-\mathcal{C}} + \mathbb{E}\|\widetilde{\boldsymbol{\lambda}}_i\|^2_{\mathcal{D}} \tag{70}$$

Following a similar argument as before, but accounting for the dual variable, we can bound:

$$\begin{bmatrix} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{c,i}\|^2 \\ \Delta_i \end{bmatrix} \leq \Gamma \begin{bmatrix} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{c,i-1}\|^2 \\ \Delta_{i-1} \end{bmatrix} + \begin{bmatrix} \mu^2\sigma^2 \\ \mu^2 b \end{bmatrix} \tag{71}$$

where

$$\Gamma = \begin{bmatrix} \sqrt{\lambda}+O(\mu^2) & O\left(\frac{\mu}{K\nu}\right) \\ O\left(\frac{\mu^2\|\mathcal{D}\|}{1-\alpha}\right) & \bar{\alpha} \end{bmatrix} \tag{72}$$

$$\sigma^2 = \frac{3}{K^2}\sum_{k=1}^{K}\beta_k^2\,\|w_k^o - w^o\|^2 + \frac{1}{K^2}\sum_{k=1}^{K}\sigma_k^2 \tag{73}$$

$$b = \|\mathcal{D}\|K^2\sigma^2 \tag{74}$$

Notably, $\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2 = 0$ no longer appears in b.

Convergence of the Universal Decentralized Algorithm

Theorem (Mean-square-behavior of the Universal Decentralized Algorithm)

Suppose all conditions of Theorem 1 hold, and additionally $\bar{\mathcal{A}}$, $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$ and μ are chosen such that $\rho(\Gamma) < 1$. Then, the iterates generated by the UDA recursions (58)–(60) satisfy:

$$\begin{bmatrix} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{c,i}\|^2 \\ \mathbb{E}\|\,\boldsymbol{\mathcal{W}}_i - \mathbb{1}\otimes \boldsymbol{w}_{c,i}\,\|^2 \end{bmatrix} \preceq O(\rho(\Gamma)^i) + \begin{bmatrix} O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 b}{K(1-\bar{\alpha})}\right) \\ O\left(\frac{\mu^2 b}{1-\bar{\alpha}}\right) + O\left(\frac{\mu^3\sigma^2}{(1-\alpha)(1-\bar{\alpha})}\right) \end{bmatrix} \tag{75}$$

where $\preceq$ denotes an elementwise inequality. Assuming μ is small enough so that we can disregard terms of order 3 or higher, it follows that:

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2 \leq O(\rho(\Gamma)^i) + O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\|\mathcal{D}\|\sigma^2}{1-\bar{\alpha}}\right) \tag{76}$$

Performance Bounds for Specific Decentralized Algorithms

Table: Steady-state performance bounds for decentralized algorithms up to second-order terms in the step-size μ obtained by specializing Theorem 2.

Algorithm	$\limsup_{i\to\infty} \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2$
DGD	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\sigma^2}{1-\lambda_2(A)}\right) + O\left(\frac{\mu^2\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2}{K(1-\lambda_2(A))^2}\right)$
Diffusion	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2^2(A)\sigma^2}{1-\lambda_2(A)}\right) + O\left(\frac{\mu^2\lambda_2^2(A)\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2}{K(1-\lambda_2(A))^2}\right)$
EXTRA	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\sigma^2}{1-\lambda_2(A)}\right)$
Exact diffusion	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2(A)\sigma^2}{1-\lambda_2(A)}\right)$
DIGing	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\sigma^2}{1-\lambda_2(A)}\right)$
NEXT	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\sigma^2}{1-\lambda_2(A)}\right)$
Aug-DGM	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2^2(A)\sigma^2}{1-\lambda_2(A)}\right)$

Observations and Take-Aways

We now present a number of observations and take-aways that arise from the performance bounds described in Table 2. Two important facts should be noted: First, the expressions only represent *bounds* on the limiting performance of the various decentralized algorithms. Second, bounds are presented in terms of $O(\cdot)$ expressions, and hence omit the dependence on certain constants. As such, some care needs to be taken when interpreting these bounds. Nevertheless, provided the bounds are sufficiently tight, and constants are consistent across expressions, meaningful interpretations can be inferred from the performance bounds. To verify this, we complement the discussion in this section by numerical results. Specifically, we will provide a collection of K agents with data following the linear model:

$$\boldsymbol{\gamma}_k = \boldsymbol{h}_k^{\mathsf{T}} w_k^o + \boldsymbol{v}_k \tag{77}$$

with isotropic regressors $\boldsymbol{h}_k \sim \mathcal{N}(0, \sigma_h^2 I_M) \in \mathbb{R}^M$ and Gaussian noise $\boldsymbol{v}_k \sim \mathcal{N}(0, \sigma_v^2) \in \mathbb{R}$.

Controlling Heterogeneity

To control the heterogeneity of the local models w_k^o, we sample them from the distribution $\mathcal{N}(\mathbb{1}, \sigma_w^2 I_M) \in \mathbb{R}^M$. In this manner, by setting $\sigma_w^2 = 0$, we recover a homogeneous data setting with $w_k^o = w^o = \mathbb{1}$, while $\sigma_w^2 > 0$ results in heterogeneous models with variance defined by σ_w^2. Each local loss is given by:

$$J_k(w) = \frac{1}{2}\mathbb{E}\|\boldsymbol{\gamma}_k - \boldsymbol{h}_k^{\mathsf{T}} w\|^2 \tag{78}$$

It can then be verified that $w_k^o = \arg\min_w J_k(w)$ and $w^o = \arg\min_w J(w)$, where $J(w) = \frac{1}{K}\sum_{k=1}^K J_k(w)$. Each agent constructs local gradient approximations:

$$\widehat{\nabla J_k}(\boldsymbol{w}_{k,i-1}) = -\boldsymbol{h}_{k,i}\left(\boldsymbol{\gamma}_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}}\,\boldsymbol{w}_{k,i-1}\right) \tag{79}$$

Algorithms (1)

We include in simulations three algorithms to represent the three families: the diffusion algorithm as a penalty-based algorithm, the Exact diffusion algorithm as a primal-dual algorithm and the Aug-DGM algorithm as an example of a gradient-tracking based algorithm. The diffusion recursions for the choice (79) amount to:

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} + \boldsymbol{h}_{k,i}\left(\gamma_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}}\,\boldsymbol{w}_{k,i-1}\right) \tag{80}$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{\psi}_{\ell,i} \tag{81}$$

The Exact diffusion recursions amount to:

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} + \boldsymbol{h}_{k,i}\left(\gamma_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}}\,\boldsymbol{w}_{k,i-1}\right) \tag{82}$$

$$\boldsymbol{\phi}_{k,i} = \boldsymbol{\psi}_{k,i} + \boldsymbol{w}_{k,i-1} - \boldsymbol{\psi}_{k,i-1} \tag{83}$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{\phi}_{\ell,i} \tag{84}$$

Algorithms (2)

For the Aug-DGM algorithm we find:

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu \boldsymbol{g}_{k,i-1} \tag{85}$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \tag{86}$$

$$\boldsymbol{g}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \left(\boldsymbol{g}_{\ell,i-1} - \boldsymbol{h}_{k,i} \left(\boldsymbol{\gamma}_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}} \boldsymbol{w}_{k,i} \right) + \boldsymbol{h}_{k,i-1} \left(\boldsymbol{\gamma}_{k,i-1} - \boldsymbol{h}_{k,i-1}^{\mathsf{T}} \boldsymbol{w}_{k,i-1} \right) \right) \tag{87}$$

We generate an Erdos-Renyi graph with edge probability $0 < p_{\text{link}} < 1$, which allows us control the level of connectivity of the graph. We construct an adjacency matrix using the Metropolis-Hastings rule (recall Lecture 3). The choice of p_{link} indirectly controls the resulting mixing rate $\lambda_2(A)$.

Baseline Simulation

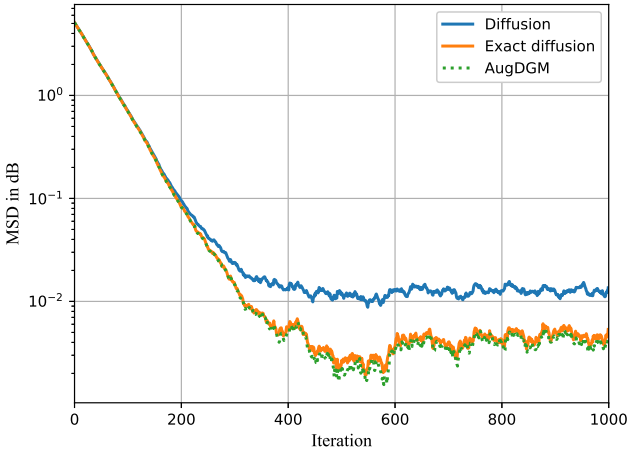


Figure: Baseline simulation with $\mu = 0.01$, $\sigma_h^2 = \sigma_v^2 = \sigma_w^2 = 1$, $p_{\text{link}} = 0.1$ and $\lambda_2(A) = 0.945$.

Observation #1: First-order expressions match

Our first observation is that the first-order terms in all performance bounds in Table 2 coincide. This indicates that for small step-sizes, as $O(\mu)$ terms dominate higher-order terms, the difference in performance between penalty-based and bias-corrected methods should decrease. This is one the next slide by reducing the step-size by a factor of 10.

Algorithm	$\limsup_{i\to\infty} \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2$
Diffusion	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2^2(A)\sigma^2}{1-\lambda_2(A)}\right) + O\left(\frac{\mu^2\lambda_2^2(A)\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2}{K(1-\lambda_2(A))^2}\right)$
Exact diffusion	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2(A)\sigma^2}{1-\lambda_2(A)}\right)$
Aug-DGM	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2^2(A)\sigma^2}{1-\lambda_2(A)}\right)$

Observation #1: First-order expressions match

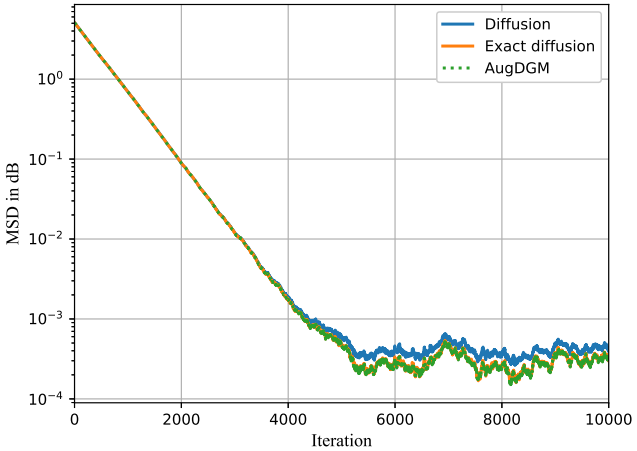


Figure: Reduced step-size with $\mu = 0.001$, $\sigma_h^2 = \sigma_v^2 = \sigma_w^2 = 1$, $p_{\text{link}} = 0.1$ and $\lambda_2(A) = 0.945$.

Observation #2: Difference is amplified for heterogeneous environments

Comparing the bounds for the diffusion, Exact diffusion and Aug-DGM, we note that the primary difference is in the term containing $\frac{1}{K}\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2$. This indicates that the difference in performance should vanish for homogeneous objectives, and is verified on the next slide by setting the variance of local models to zero, i.e., $\sigma_w^2 = 0$.

Algorithm	$\limsup_{i\to\infty} \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2$
Diffusion	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2^2(A)\sigma^2}{1-\lambda_2(A)}\right) + O\left(\frac{\mu^2\lambda_2^2(A)\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2}{K(1-\lambda_2(A))^2}\right)$
Exact diffusion	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2(A)\sigma^2}{1-\lambda_2(A)}\right)$
Aug-DGM	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2^2(A)\sigma^2}{1-\lambda_2(A)}\right)$

Observation #2: Difference is amplified for heterogeneous environments

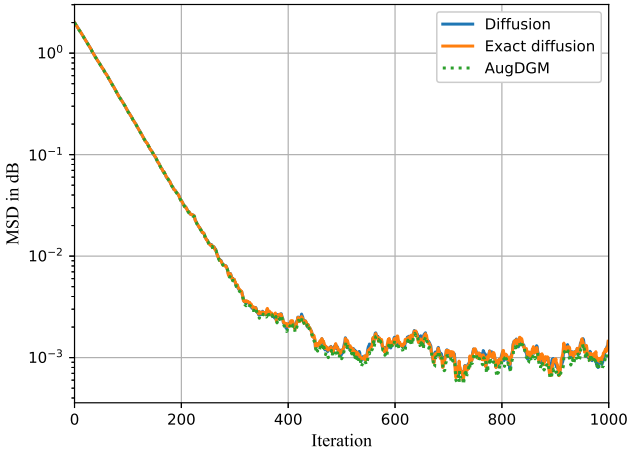


Figure: A homogeneous setting with $\mu = 0.01$, $\sigma_h^2 = \sigma_v^2 = 1$, $\sigma_w^2 = 0$, $p_{\text{link}} = 0.1$ and $\lambda_2(A) = 0.945$.

Observation #3: Difference is reduced in densely connected networks

Our third observation is that the term distinguishing the performance of the three algorithms is multiplied by $\frac{1}{(1-\lambda_2(A))^2}$. This indicates that the difference in performance is reduced for densely connected networks, where $\lambda_2(A) \to 0$. This is verified on the next slide by increasing the connectivity via $p_{\text{link}} = 0.5$, which results in $\lambda_2(A) = 0.725$.

Algorithm	$\limsup_{i\to\infty} \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2$
Diffusion	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K \lambda_2^2(A)\sigma^2}{1-\lambda_2(A)}\right) + O\left(\frac{\mu^2\lambda_2^2(A)\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2}{K(1-\lambda_2(A))^2}\right)$
Exact diffusion	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K \lambda_2(A)\sigma^2}{1-\lambda_2(A)}\right)$
Aug-DGM	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K \lambda_2^2(A)\sigma^2}{1-\lambda_2(A)}\right)$

Observation #3: Difference is amplified in sparsely connected networks

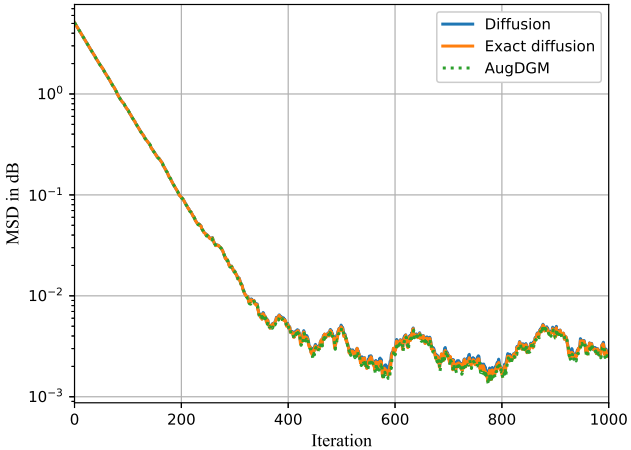


Figure: A well-connected graph with $\sigma_h^2 = \sigma_v^2 = \sigma_w^2 = 1$, $p_{\text{link}} = 0.5$ and $\lambda_2(A) = 0.725$.

Observation #4: Incremental variants exhibit additional variance reduction

Our final observation is that second-order terms in the incremental methods, compared to their non-incremental counterparts, exhibit additional variance reduction by a factor of $\lambda_2(A)$ or $\lambda_2^2(A)$. We illustrate this on the next slide by comparing the performance of the diffusion algorithm with that of the DGD algorithm. We observe that the diffusion strategy outperforms the DGD algorithm by a small margin.

Algorithm	$\limsup_{i\to\infty} \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2$
DGD	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\sigma^2}{1-\lambda_2}\right) + O\left(\frac{\mu^2\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2}{K(1-\lambda_2)^2}\right)$
Diffusion	$O\left(\mu\sigma^2\right) + O\left(\frac{\mu^2 K\lambda_2^2(A)\sigma^2}{1-\lambda_2(A)}\right) + O\left(\frac{\mu^2\lambda_2^2(A)\sum_{k=1}^{K}\|\nabla J_k(w^o)\|^2}{K(1-\lambda_2(A))^2}\right)$

Observation #4: Incremental variants exhibit additional variance reduction

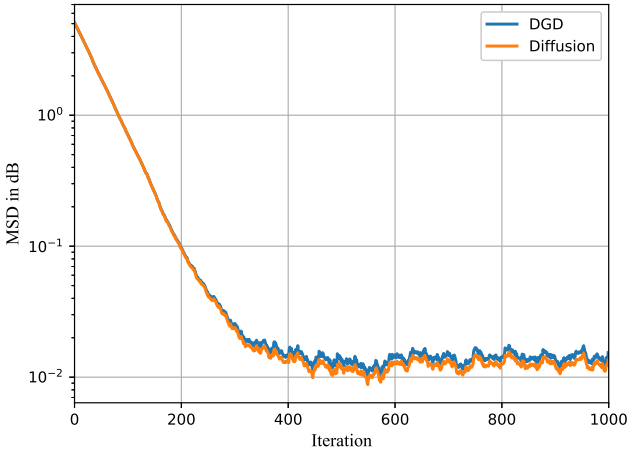


Figure: Comparison of DGD (non-incremental) and diffusion (incremental) algorithms with $\sigma_h^2 = \sigma_v^2 = \sigma_w^2 = 1$, $p_{\text{link}} = 0.1$ and $\lambda_2(A) = 0.945$.

Stability of Incremental Variants

A related fact to our previous observation is that incremental variants of decentralized algorithms tend to have wider stability ranges than their non-incremental counterparts. This is due to an asymmetry present in non-incremental recursions. We can illustrate this by comparing the DGD:

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k}\, \boldsymbol{w}_{\ell,i-1} - \mu \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{88}$$

and diffusion recursions:

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \left(\boldsymbol{w}_{\ell,i-1} - \mu \widehat{\nabla J}_\ell(\boldsymbol{w}_{\ell,i-1}) \right) \tag{89}$$

We verify this by on the next slide by increasing the step-size of until one of the algorithms becomes unstable.

Stability of Incremental Variants

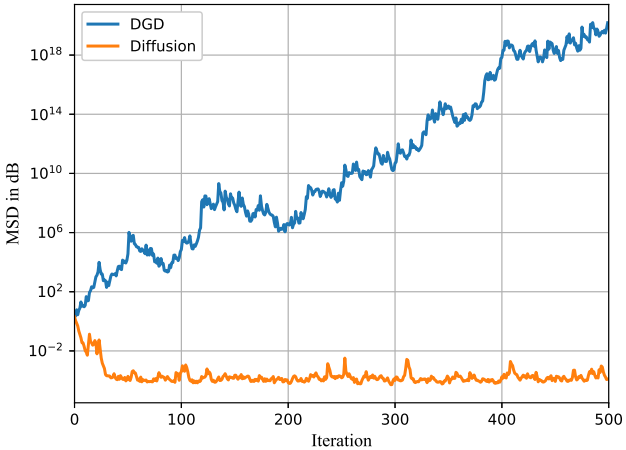


Figure: Comparison of DGD (non-incremental) and diffusion (incremental) algorithms with $\mu = 0.75$, $\sigma_h^2 = 1$, $\sigma_v^2 = 0.01$,$\sigma_w^2 = 0$, $p_{\text{link}} = 0.1$ and $\lambda_2(A) = 0.945$.

Conclusion

- We presented performance bounds for penalty-based (DGD and diffusion), primal-dual (EXTRA and Exact diffusion) and gradient-tracking (DIGing, NEXT, Aug-DGM) algorithms.
- To first order in the step-size, performance bounds are identical amongst all algorithms, and match those of centralized architectures.
  - This implies that for homogeneous objectives and small step-sizes, all algorithms exhibit linear performance gain.
- Bias-corrected algorithms outperform penalty-based ones particularly for:
  - Moderately large step-sizes
  - Sparse graphs
  - Heterogeneous objectives
  - Exact gradients or very low gradient noise
- Incremental algorithms exhibit better performance and wider stability ranges than their non-incremental counterparts.

References and Further Reading

- General references and surveys:
  - A. H. Sayed, *Inference and Learning from Data*, Cambridge University Press, 2022.
  - A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, July 2014.
  - A. H. Sayed, "Adaptive Networks," in *Proceedings of the IEEE*, vol. 102, no. 4, pp. 460-497, 2014.
  - S. Vlaski, S. Kar, A. H. Sayed and J. M. F. Moura, "Networked Signal and Information Processing: Learning by multiagent systems," in *IEEE Signal Processing Magazine*, vol. 40, no. 5, pp. 92-105, July 2023,
- Performance guarantees:
  - J. Chen and A. H. Sayed, "On the Learning Behavior of Adaptive Networks—Part I: Transient Analysis," in *IEEE Trans. Information Theory*, vol. 61, no. 6, pp. 3487-3517, June 2015.
  - S. A. Alghunaim, A. H. Sayed, "Linear convergence of primal-dual gradient methods and their performance in distributed optimization," *Automatica*, vol. 117, 2020.
  - K. Yuan, S. A. Alghunaim, B. Ying and A. H. Sayed, "On the Influence of Bias-Correction on Distributed Stochastic Optimization," in *IEEE Trans. Signal Processing*, vol. 68, pp. 4352-4367, 2020.