Part II

Distributed Learning with a Fusion Center

4 Centralized Learning

As we have seen in previous chapters, a large number of inference and learning tasks can be formulated as optimization problems over a set of parameters w. In Chapters 2 and 3 we showed how first-order algorithms, such as gradient descent and its stochastic variants, can be used to pursue optimal models systematically and with performance guarantees. With the exception of the case study in Chapter 1, all of these techniques were based on single agents, with a single set of data, optimizing a single objective function. We will now show how first-order methods can be adapted to multi-agent systems, where we are interested in finding models by fitting to data from multiple data sources.

4.1 CONSENSUS OPTIMIZATION

Let us recall the empirical risk minimization problem:

$$w_k^\star = \arg\min_w J_k(w) = \arg\min_w \frac{1}{N}\sum_{n=1}^{N} Q(w; x_{k,n}) \tag{4.1}$$

Here, w denote model parameters, $x_{k,n}$ denotes the n-th sample available to agent k, and $Q(w; x_{k,n})$ quantifies the fit of model w to the data $x_{k,n}$. The model $w_k^\star$ is then optimal based on the data available to agent k. We can pursue $w_k^\star$ using the gradient descent algorithm studied in Chapter 2

$$w_i = w_{i-1} - \mu\nabla J_k(w_{i-1}) = w_{i-1} - \frac{1}{N}\sum_{n=1}^{N}\nabla Q(w_{i-1}; x_{k,n}) \tag{4.2}$$

or its stochastic variants introduced in Chapter 3. Instead of pursuing locally optimal models, we can instead define a globally optimal model:

$$w^\star = \arg\min_w J(w) = \arg\min_w \frac{1}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N} Q(w; x_{k,n}) \tag{4.3}$$

In defining $J(w)$, we are now averaging the loss $Q(w; x_{k,n})$ over both the agent index k and the sample index n, hence aggregating all data across the network. For this reason we refer to $w^\star$ as the globally optimal model. Comparing the

local and global objectives (4.1) and (4.3), we observe the useful relationship:

$$\boxed{J(w) = \frac{1}{K}\sum_{k=1}^{K} J_k(w)} \tag{4.4}$$

We refer to problem (4.3), which is equivalent to (4.4), as the *consensus optimization problem*, since we are looking for a single model w that fits data across the entire collection of agents optimally. In the absence of constraints on the exchange of information, we can apply gradient descent directly to the consensus problem (4.4) and develop the recursion:

$$\begin{aligned} w_i &= w_{i-1} - \mu\nabla J(w_{i-1}) \\ &= w_{i-1} - \frac{\mu}{K}\sum_{k=1}^{K}\nabla J_k(w_{i-1}) \\ &= w_{i-1} - \frac{\mu}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N}\nabla Q(w_{i-1};x_{k,n}) \end{aligned} \tag{4.5}$$

Recursion (4.5) provides an algorithm for solving the consensus optimization problem, but requires central aggregation of the raw data $x_{k,n}$ in order to compute the aggregate gradient $\frac{1}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N}\nabla Q(w_{i-1};x_{k,n})$.

4.1.1 Consensus Optimization for Expected Risk Minimization

Our discussion leading to the consensus optimization (4.4) was focused on *empirical risk* minimization problems of the form (4.1) and (4.3). Empirical risk minimization problems are based on a finite batch of data. As we saw in Chapters 1 and 3, we may also be interested in local objectives which take the form of an expected risk:

$$J_k(w) = \mathbb{E}_{\boldsymbol{x}_k} Q(w;\boldsymbol{x}_k) \tag{4.6}$$

Recall from Example 3.2 that (4.6) can be viewed as a generalization of the empirical risk minimization problem (4.1). Indeed, if we define:

$$\boldsymbol{x}_k = \begin{cases} x_{k,1}, & \text{with probability } \frac{1}{N}, \\ x_{k,2}, & \text{with probability } \frac{1}{N}, \\ \vdots & \\ x_{k,N}, & \text{with probability } \frac{1}{N}. \end{cases} \tag{4.7}$$

we can verify that (4.6) reduces to (4.1).

In the case of empirical risk minimization problems, the consensus problem (4.4) carries a clear interpretation as the aggregate empirical risk the the losses are averaged globally across all data available at all agents. Analogously

to (4.4), we can define a consensus problem for expected local risks (4.6):

$$J(w) = \frac{1}{K}\sum_{k=1}^{K} J_k(w) = \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}_{\boldsymbol{x}_k} Q(w;\boldsymbol{x}_k) \tag{4.8}$$

To develop an interpretation for (4.8), we introduce a random variable $\boldsymbol{x}$ as a mixture of the local data $\boldsymbol{x}_k$ as:

$$\boldsymbol{x} = \begin{cases} \boldsymbol{x}_1, & \text{with probability } \frac{1}{K}, \\ \boldsymbol{x}_2, & \text{with probability } \frac{1}{K}, \\ \vdots & \\ \boldsymbol{x}_K, & \text{with probability } \frac{1}{K}. \end{cases} \tag{4.9}$$

We can then verify that:

$$\begin{aligned} J(w) &= \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}_{\boldsymbol{x}_k} Q(w;\boldsymbol{x}_k) \\ &= \sum_{k=1}^{K} \Pr\{\boldsymbol{x}_k = \boldsymbol{x}\} \cdot \mathbb{E}_{\boldsymbol{x}} \{Q(w;\boldsymbol{x}) | \boldsymbol{x} = \boldsymbol{x}_k\} \\ &= \mathbb{E}_{\boldsymbol{x}} Q(w;\boldsymbol{x}) \end{aligned} \tag{4.10}$$

In other words, the consensus optimization problem (4.8) when applied to expected local risks (4.6) can be interpreted as a global expected risk minimization problem applied to a global random variable $\boldsymbol{x}$, which takes the form of a mixture of the local data distributions according to (4.9). We conclude that even in the more general case of expected risk minimization, the solution of consensus optimization problems is meaningful.

4.2 DISTRIBUTED PARADIGMS

The current as well as several next chapters will develop algorithms for solving consensus problems of the form (4.4) while avoiding central aggregation and processing of raw data. We will refer to all of these algorithms as **distributed algorithms**, because in contrast to (4.5), data and computations will be distributed over agents in a network. We will encounter a number of different learning paradigms with different restrictions on computations and communication, which we preview here, in order to provide context for subsequent discussions.

Non-cooperative learning: In non-cooperative paradigms, individual agents solve independent learning problems of the form (4.1) via local recursions of the form (4.2) or their variations discussed in Chapters 2 and 3. These algorithms are simple and require no communication, but can only guarantee *local optimality*

in the sense of (4.1).

Centralized or parallel learning: Centralized learning paradigms involve a fusion center, which can also be referred to as a parameter server. Agents communicate directly with the fusion center, and exchange processed iterates in lie of raw data at regular intervals. While a bulk of computations are carried out on individual agents, the fusion center plays the role of an orchestrator, and coordinates the learning process by aggregating local estimates. Due to their need for regular and reliable exchanges, centralized architectures are most suited in well-controlled networks, such as a server cluster. We will develop and study centralized learning algorithms in this chapter.

Federated learning: Federated architectures also involve a fusion center, which plays the role of a coordinator. In contrast to centralized or parallel architectures, federated structures allow for high levels asynchrony, irregularity and imperfections in agent participation, computation and communication. As such, federated structures are more appropriate in uncontrolled multi-agents systems, such as mobile devices, autonomous vehicles, or unreliable sensor networks. We will study federated learning paradigms in future chapters.

Decentralized learning: Decentralized learning paradigms completely avoid the need for central aggregation, and instead rely only on peer-to-peer interactions between nearby agents. The flow of information is governed by an underlying network topology, which limits which agents can interact with one another. The lack of a central coordinator complicates the resulting learning dynamics, but on the other hand decentralized systems have certain advantages when it comes to considerations around privacy, robustness to node and link failure, and communication efficiency. We will study decentralized learning paradigms in future chapters.

Different paradigms are visualized in Fig. 4.1.

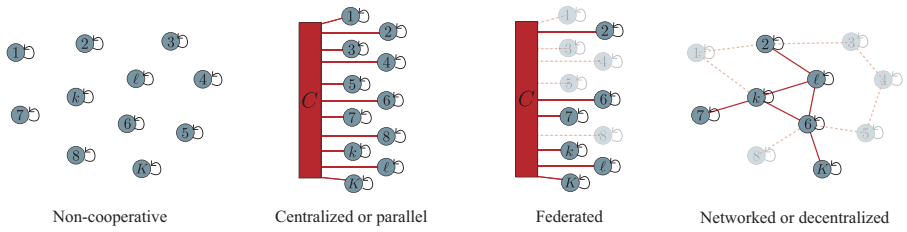

Figure 4.1 Taxonomy of paradigms for distributed learning.

4.3 CENTRALIZED GRADIENT DESCENT

We begin by developing a distributed variant of the gradient descent algorithm studied in Chapter 2. Applying gradient descent directly to the consensus problem (4.4), we obtain (4.5), repeated here for reference:

$$w_i = w_{i-1} - \frac{\mu}{K}\sum_{k=1}^{K} \nabla J_k(w_{i-1}) \tag{4.11}$$

While it is not distributed by default, the gradient recursion is amenable to a distributed implementation in a straightforward manner. We may devise two distinct scheme:

Parameter exchange: The fusion center sends the current model w_{i-1} to all agents. Then, each agent performs a local update by descending along its own local cost function based on its private data:

$$\psi_{k,i} = w_{i-1} - \mu\nabla J_k(w_{i-1}) \tag{4.12}$$

The locally updated models are sent back to the fusion center, where they are aggregated according to:

$$w_i = \frac{1}{K}\sum_{k=1}^{K} \psi_{k,i} \tag{4.13}$$

Putting (4.12) and (4.13) together, we can verify that:

$$w_i = \frac{1}{K}\sum_{k=1}^{K} (w_{i-1} - \mu\nabla J_k(w_{i-1})) = w_{i-1} - \frac{\mu}{K}\sum_{k=1}^{K} \nabla J_k(w_{i-1}) \tag{4.14}$$

and hence (4.12)–(4.13) generates the same sequence of iterates w_i without the need to exchange any data. Instead, agents and the parameter serve pass back and forth prior global models w_{i-1} and intermediate local models $\psi_{k,i}$, which contain sufficient information to implement the gradient descent recursion (4.11).

Gradient exchange: As an alternative to the distributed implementation (4.12)–(4.13), we can implement (4.11) in a distributed manner by exchanging gradients instead of models. In this setting, at iteration i, the parameter server sends the prior model w_{i-1} to all agents. Each agent computes the local gradient, evaluated at the model w_{i-1}:

$$g_{k,i} = \nabla J_k(w_{i-1}) \tag{4.15}$$

Each agent then sends back the local gradient to the parameter server, where the update is computed via:

$$w_i = w_{i-1} - \frac{\mu}{K}\sum_{k=1}^{K} g_{k,i} \tag{4.16}$$

It is again straightforward to verify that the resulting recursion is equivalent to (4.11).

We conclude from these implementations, that the intermediate model $\psi_{k,i}$ in (4.12) or the local gradient $g_{k,i}$ (4.15) contain a sufficient summary of the local cost and data sets for the purposes of implementing the centralized gradient descent recursion (4.11) in a distributed manner.

Example 4.1 (**Logistic regression**) We consider a distributed classification problem, where each local cost is giving by a logistic regression problem:

$$J_k(w) = \frac{\rho}{2}\|w\|^2 \;+\; \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{-\gamma_{k,n} h_{k,n}^{\mathsf{T}} w}\right) \tag{4.17}$$

We may then pursue a solution to the consensus problem:

$$J(w) = \frac{\rho}{2}\|w\|^2 \;+\; \frac{1}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N} \ln\left(1 + e^{-\gamma_{k,n} h_{k,n}^{\mathsf{T}} w}\right) \tag{4.18}$$

in a distributed manner. For example, the parameter-based distributed implementation of centralized gradient descent (4.12)–(4.13) takes the form:

$$\psi_{k,i} = (1-\mu\rho)w_{i-1} + \mu\left(\frac{1}{N}\sum_{n=1}^{N} \frac{\gamma_{k,n} h_{k,n}}{1 + e^{\gamma_{k,n} h_{k,n}^{\mathsf{T}} w_{i-1}}}\right) \tag{4.19}$$

$$w_i = \frac{1}{K}\sum_{k=1}^{K} \psi_{k,i} \tag{4.20}$$

4.3.1 Convergence of Centralized Gradient Descent

As we argued in Sec. 4.3, the recursions (4.11), (4.12)–(4.13) and (4.15)–(4.16) all generate identical iterates w_i and correspond to gradient descent recursions on the consensus optimization problem $\frac{1}{K}\sum_{k=1}^{K} J_k(w)$. It then follows that we can directly apply Theorem 2.1 to conclude that the iterates w_i will converge to:

$$w^\star \triangleq \arg\min_{w} \frac{1}{K}\sum_{k=1}^{K} J_k(w) \tag{4.21}$$

linearly, i.e.:

$$\|w^\star - w_i\|^2 \;\leq\; \lambda\,\|w^\star - w_{i-1}\|^2 \tag{4.22}$$

where $\lambda \triangleq 1 - 2\mu\nu + \mu^2\delta^2$ and ν and δ correspond to the strong-convexity and smoothness constants of the aggregate cost $J(w)$.

4.4 CENTRALIZED STOCHASTIC GRADIENT DESCENT

As we motivated in Chapter 3, in many inference and learning applications, we are unable to compute exact gradients $\nabla J_k(w)$, either due to lack of computational capabilities, or due to the absence of distributional information about the underlying data. To account for this fact, we will now generalize the centralized deterministic gradient descent algorithm of Sec. 4.3 to allow for gradient approximations. We continue to study consensus optimization problems of the form:

$$w^o \triangleq \arg\min_w \frac{1}{K}\sum_{k=1}^{K} J_k(w) \tag{4.23}$$

where the local objectives now take the form of a general expected risk:

$$J_k(w) = \mathbb{E}_{\boldsymbol{x}_k} Q(w; \boldsymbol{x}_k) \tag{4.24}$$

We begin by applying stochastic gradient descent (see Chapter 3) to (4.23). Given local gradient approximations $\widehat{\nabla J_k}(w)$, we may construct an approximation of the global gradient $\nabla J(w)$ as:

$$\widehat{\nabla J}(w) = \frac{1}{K}\sum_{k=1}^{K} \widehat{\nabla J_k}(w) \tag{4.25}$$

Suppose each local gradient approximation $\widehat{\nabla J_k}(w)$ satisfies the gradient noise conditions (**??**)–(3.23). For simplicity, we restrict ourselves in this chapter to gradient approximations with $\alpha^2 = \gamma^2 = 0$ resulting in:

$$\mathbb{E}\left\{\widehat{\nabla J_k}(\boldsymbol{w}_{i-1})|\boldsymbol{w}_{i-1}\right\} = \nabla J_k(\boldsymbol{w}_{i-1}) \tag{4.26}$$

$$\mathbb{E}\left\{\left\|\widehat{\nabla J_k}(\boldsymbol{w}_{i-1}) - \nabla J_k(\boldsymbol{w}_{i-1})\right\|^2 |\boldsymbol{w}_{i-1}\right\} \leq \beta_k^2 \|w_k^o - \boldsymbol{w}_{i-1}\|^2 + \sigma_k^2 \tag{4.27}$$

We index the noise constants β_k^2, σ_k^2 by k to emphasize the fact that different agents may have accesss to gradient approximations of varying quality. Note that the relative component $\beta_k^2\|w_k^o - \boldsymbol{w}_{i-1}\|^2$ is measured relative to w_k^o, which denotes the local minimizer:

$$w_k^o \triangleq \arg\min_w J_k(w) \tag{4.28}$$

We would then like to establish the accuracy of $\widehat{\nabla J}(w)$ in (4.25). First, to verify that (4.25) is unbiased, we note that:

$$\begin{aligned}\mathbb{E}\left\{\widehat{\nabla J}(\boldsymbol{w}_{i-1})|\boldsymbol{w}_{i-1}\right\} &= \mathbb{E}\left\{\frac{1}{K}\sum_{k=1}^{K}\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})|\boldsymbol{w}_{i-1}\right\} \\ &= \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\left\{\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})|\boldsymbol{w}_{i-1}\right\} \\ &\overset{(4.26)}{=} \frac{1}{K}\sum_{k=1}^{K}\nabla J_k(\boldsymbol{w}_{i-1}) \end{aligned} \tag{4.29}$$

For the variance, we have:

$$\begin{aligned}&\mathbb{E}\left\{\left\|\widehat{\nabla J}(\boldsymbol{w}_{i-1})-\nabla J(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\} \\ &= \mathbb{E}\left\{\left\|\frac{1}{K}\sum_{k=1}^{K}\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\frac{1}{K}\sum_{k=1}^{K}\nabla J_k(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\} \\ &= \frac{1}{K^2}\sum_{k=1}^{K}\mathbb{E}\left\{\left\|\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J_k(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\} \\ &\quad+\sum_{k=1}^{K}\sum_{\ell\neq k}\mathbb{E}\left\{\left(\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J_k(\boldsymbol{w}_{i-1})\right)^{\mathsf{T}}\left(\widehat{\nabla J}_\ell(\boldsymbol{w}_{i-1})-\nabla J_\ell(\boldsymbol{w}_{i-1})\right)|\boldsymbol{w}_{i-1}\right\} \\ &\overset{(a)}{=} \frac{1}{K^2}\sum_{k=1}^{K}\mathbb{E}\left\{\left\|\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J_k(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\} \\ &\quad+\sum_{k=1}^{K}\sum_{\ell\neq k}\mathbb{E}\left\{\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J_k(\boldsymbol{w}_{i-1})|\boldsymbol{w}_{i-1}\right\}^{\mathsf{T}}\mathbb{E}\left\{\widehat{\nabla J}_\ell(\boldsymbol{w}_{i-1})-\nabla J_\ell(\boldsymbol{w}_{i-1})|\boldsymbol{w}_{i-1}\right\} \\ &\overset{(4.26)}{=} \frac{1}{K^2}\sum_{k=1}^{K}\mathbb{E}\left\{\left\|\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J_k(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\} \\ &\overset{(4.27)}{\leq} \frac{1}{K^2}\sum_{k=1}^{K}\left(\beta_k^2\|w_k^o-\boldsymbol{w}_{i-1}\|^2+\sigma_k^2\right)\end{aligned} \tag{4.30}$$

where (a) follows as long as local gradient approximations are independent. To bring this relation into a form that we can use in Theorem 3.1, we further refor-

mulate:

$$\begin{aligned}
&\mathbb{E}\left\{\left\|\widehat{\nabla J}(\boldsymbol{w}_{i-1})-\nabla J(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\}\\
&\overset{(a)}{\leq}\frac{1}{K^2}\sum_{k=1}^{K}\left(\beta_k^2\|w_k^o-w^o+w^o-\boldsymbol{w}_{i-1}\|^2+\sigma_k^2\right)\\
&\overset{(b)}{\leq}\frac{1}{K^2}\sum_{k=1}^{K}\left(2\beta_k^2\|w_k^o-w^o\|^2+2\beta_k^2\|w^o-\boldsymbol{w}_{i-1}\|^2+\sigma_k^2\right)\\
&=\frac{1}{K^2}\left(\sum_{k=1}^{K}2\beta_k^2\right)\|w^o-\boldsymbol{w}_{i-1}\|^2+\frac{1}{K^2}\sum_{k=1}^{K}\left(2\beta_k^2\|w_k^o-w^o\|^2+\sigma_k^2\right)\\
&\overset{(c)}{=}\beta^2\|w^o-\boldsymbol{w}_{i-1}\|^2+\sigma^2
\end{aligned}\tag{4.31}$$

In (a) we added and subtracted w^o, (b) follows from the relation $\|a+b\|^2\leq 2\|a\|^2+2\|b\|^2$ for any pair of vectors a,b and in step (c) we defined:

$$\beta^2=\frac{1}{K^2}\left(\sum_{k=1}^{K}2\beta_k^2\right)\tag{4.32}$$

$$\sigma^2=\frac{1}{K^2}\sum_{k=1}^{K}\left(2\beta_k^2\|w_k^o-w^o\|^2+\sigma_k^2\right)\tag{4.33}$$

We conclude that the average of local gradient approximations (4.25) is a valid gradient approximation of the global gradient $\nabla J(w)$, and we can hence formulate the stochastic gradient recursions:

$$\boldsymbol{w}_i=\boldsymbol{w}_{i-1}-\mu\widehat{\nabla J}(\boldsymbol{w}_{i-1})=\boldsymbol{w}_{i-1}-\frac{\mu}{K}\sum_{k=1}^{K}\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})\tag{4.34}$$

Analogously to the centralized gradient algorithm with exact gradients, we can again distribute (4.34) in one of two ways:

Parameter exchange: The fusion center sends the current model $\boldsymbol{w}_{i-1}$ to all agents. Each agent then performs a local update using the gradient approximation of its local cost function:

$$\boldsymbol{\psi}_{k,i}=\boldsymbol{w}_{i-1}-\mu\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})\tag{4.35}$$

The locally updated models are sent back to the fusion center, where they are aggregated according to:

$$\boldsymbol{w}_i=\frac{1}{K}\sum_{k=1}^{K}\boldsymbol{\psi}_{k,i}\tag{4.36}$$

It can then be verified that (4.35)–(4.36) is equivalent to (4.34), while requiring only the exchange of intermediate models $\boldsymbol{\psi}_{k,i}$.

Gradient exchange: We can also implement (4.34) in a distributed manner by instead exchanging gradient approximations. Again, at iteration i, the parameter server sends the prior model $\boldsymbol{w}_{i-1}$ to all agents. Each agent computes the local gradient, evaluated at the model $\boldsymbol{w}_{i-1}$:

$$\boldsymbol{g}_{k,i} = \widehat{\nabla J}_k(\boldsymbol{w}_{i-1}) \tag{4.37}$$

Each agent then sends back the local gradient to the parameter server, where the update is computed via:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \frac{\mu}{K}\sum_{k=1}^{K} \boldsymbol{g}_{k,i} \tag{4.38}$$

It is again straightforward to verify that the resulting recursion is equivalent to (4.34).

Example 4.2 (**Least-Mean Square**) Suppose each agent in the network observes data arising from a linear model:

$$\boldsymbol{\gamma}_k = \boldsymbol{h}_k^{\mathsf{T}} w^o + \boldsymbol{v}_k \tag{4.39}$$

Note that while the parameters w^o in this example are common to all agents, the random variables $\boldsymbol{h}_k$, $\boldsymbol{v}_k$ and hence $\boldsymbol{\gamma}_k$ may follow distinct distributions. For example, different agents may be subjected to different levels of noise $\boldsymbol{v}_k$, and may hence have observations of varying quality. We may then formulate the local expected risk problems:

$$J_k(w) = \mathbb{E}\,\|\boldsymbol{\gamma}_k - \boldsymbol{h}_k^{\mathsf{T}} w\|^2 \tag{4.40}$$

If we employ an ordinary gradient approximation of the form:

$$\widehat{\nabla J}(w) = \nabla Q(w; \boldsymbol{h}_{k,i}, \boldsymbol{\gamma}_{k,i}) = \boldsymbol{h}_{k,i}\left(\boldsymbol{\gamma}_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}} w\right) \tag{4.41}$$

This leads to a distributed implementation of the centralized stochastic gradient recursion of the form:

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{i-1} - \mu \boldsymbol{h}_{k,i}\left(\boldsymbol{\gamma}_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}} \boldsymbol{w}_{i-1}\right) \tag{4.42}$$

$$\boldsymbol{w}_i = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{\psi}_{k,i} \tag{4.43}$$

4.4.1 Performance of Centralized Stochastic Gradient Descent

Since the centralized stochastic gradient algorithms (4.34), (4.35)–(4.36) and (4.37)–(4.38) all yield identical iterates, and all correspond to stochastic gradient descent with the gradient approximation (4.25) applied to the aggregate cost $J(w)$, we

may infer the resulting performance from Theorem 3.1. With the expression for the gradient noise constants in (4.33), we find:

$$\lim_{i\to\infty} \mathbb{E}\|w^o - \boldsymbol{w}_i\|^2 \leq \frac{\mu}{\nu K^2} \sum_{k=1}^{K} \left(2\beta_k^2 \|w_k^o - w^o\|^2 + \sigma_k^2\right) \tag{4.44}$$

Linear Performance Gain

A useful simplification and take-away occurs if we consider the setting of perfectly homogenous agents. In a homogenous setting, the data $\boldsymbol{x}_k$ observed by each agent is identically distributed, and all gradient approximations take the same form. It follows that:

$$J_k(w) = \mathbb{E}_{\boldsymbol{x}_k} Q(w; \boldsymbol{x}_k) = \mathbb{E}_{\boldsymbol{x}_\ell} Q(w; \boldsymbol{w}_\ell) \tag{4.45}$$

for all k, l, and hence $J_k(w) = J(w)$ for all k. Similarly, we can conclude that $w_k^o = w^o$ and $\sigma_k^2 = \sigma^2$. Then (4.44) simplifies to:

$$\lim_{i\to\infty} \mathbb{E}\|w^o - \boldsymbol{w}_i\|^2 \leq \frac{\mu\sigma^2}{\nu K} \tag{4.46}$$

We conclude that the limiting performance of the centralized architecture improves at a rate of $\frac{1}{K}$, where K is the number of agents. This fact is referred to as *linear gain* in distributed systems and is a key motivator for agents to participate in learning protocol. It indicates that a single agent can expect improved performance by joining a cooperative learning protocol, and the level of improvement grows with the size of the network.

4.5 ASYNCHRONOUS OPTIMIZATION

In Sections 4.3 and 4.4 we developed algorithms for the deterministic and stochastic optimization of consensus optimization problems of the form (4.4). So far, however, we have assumed full participation of all agents at every iteration. It is common in practice to be faced with asynchrony in the form of intermittent communication, or unreliable computation. We will demonstrate now how the tools of stochastic approximation thoery can be leveraged in this context as well to efficiently develop algorithms and performance guarantees. We consider again the setting of Section 4.4, where we optimize:

$$J(w) = \frac{1}{K} \sum_{k=1}^{K} J_k(w) \tag{4.47}$$

with $J_k(w) = \mathbb{E}_{\boldsymbol{x}_k} Q(w; \boldsymbol{x}_k)$ based on local gradient approximations $\widehat{\nabla J}_k(w)$. Instead of incolving all agents at each iteration as in the case of (4.35)–(4.36), we now study an *asynchronous* variant where at each iteration a single agent is selected, and the parameter only interacts with the selected agent. To make this

precise, we introduce the random index $\boldsymbol{k}_i$, which denotes the agent picked at time instant i, and follows a uniform distribution:

$$\boldsymbol{k}_i = \begin{cases} 1, & \text{with probability } \frac{1}{K}, \\ 2, & \text{with probability } \frac{1}{K}, \\ \vdots & \\ K, & \text{with probability } \frac{1}{K}. \end{cases} \tag{4.48}$$

The parameter server then provides the selected agent $\boldsymbol{k}_i$ with the previous model $\boldsymbol{w}_{i-1}$. Agent $\boldsymbol{k}_i$ can then update the model:

$$\boldsymbol{\psi}_{\boldsymbol{k}_i,i} = \boldsymbol{w}_{i-1} - \mu\widehat{\nabla J}_{\boldsymbol{k}_i}(\boldsymbol{w}_{i-1}) \tag{4.49}$$

The selected agent then sends the model back to the parameter server, where the global model is updated according to:

$$\boldsymbol{w}_i = \boldsymbol{\psi}_{\boldsymbol{k}_i,i} \tag{4.50}$$

We can write (4.49)–(4.50) compactly as:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu\widehat{\nabla J}_{\boldsymbol{k}_i}(\boldsymbol{w}_{i-1}) = \boldsymbol{w}_{i-1} - \mu\widehat{\nabla J}(\boldsymbol{w}_{i-1}) \tag{4.51}$$

where we defined:

$$\widehat{\nabla J}(\boldsymbol{w}_{i-1}) = \widehat{\nabla J}_{\boldsymbol{k}_i}(\boldsymbol{w}_{i-1}) \tag{4.52}$$

In order to study the performance of this asynchronous algorithm, we hence only need to establish conditions on the gradient noise induced by the approximation (4.52). First, we have:

$$\begin{aligned} \mathbb{E}\left\{\widehat{\nabla J}(\boldsymbol{w}_{i-1})|\boldsymbol{w}_{i-1}\right\} &= \mathbb{E}\left\{\widehat{\nabla J}_{\boldsymbol{k}_i}(\boldsymbol{w}_{i-1})|\boldsymbol{w}_{i-1}\right\} \\ &\overset{(a)}{=} \sum_{k=1}^{K} \mathbb{E}\left\{\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})|\boldsymbol{k}_i = k, \boldsymbol{w}_{i-1}\right\} \cdot \Pr\{\boldsymbol{k}_i = k\} \\ &= \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}\left\{\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})|\boldsymbol{w}_{i-1}\right\} \\ &\overset{(4.26)}{=} \frac{1}{K}\sum_{k=1}^{K} \nabla J_k(\boldsymbol{w}_{i-1}) \\ &= \nabla J(\boldsymbol{w}_{i-1}) \end{aligned} \tag{4.53}$$

where (a) follows from the law of total probability. For the variance, we find:

$$
\begin{aligned}
&\mathbb{E}\left\{\left\|\widehat{\nabla J}(\boldsymbol{w}_{i-1})-\nabla J(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\}\\
&=\mathbb{E}\left\{\left\|\widehat{\nabla J}_{\boldsymbol{k}_i}(\boldsymbol{w}_{i-1})-\nabla J(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\}\\
&=\sum_{k=1}^{K}\mathbb{E}\left\{\left\|\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{k}_i=k,\boldsymbol{w}_{i-1}\right\}\cdot\Pr\{\boldsymbol{k}_i=k\}\\
&=\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\left\{\left\|\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J_k(\boldsymbol{w}_{i-1})+\nabla J_k(\boldsymbol{w}_{i-1})-\nabla J(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\}\\
&\overset{(4.26)}{=}\frac{1}{K}\sum_{k=1}^{K}\left(\mathbb{E}\left\{\left\|\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J_k(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\}+\|\nabla J_k(\boldsymbol{w}_{i-1})-\nabla J(\boldsymbol{w}_{i-1})\|^2\right)\\
&=\frac{1}{K}\sum_{k=1}^{K}\left(\mathbb{E}\left\{\left\|\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J_k(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\}+2\|\nabla J_k(\boldsymbol{w}_{i-1})\|^2+2\|\nabla J(\boldsymbol{w}_{i-1})\|^2\right)\\
&\overset{(a)}{\leq}\frac{1}{K}\sum_{k=1}^{K}\left(\mathbb{E}\left\{\left\|\widehat{\nabla J}_k(\boldsymbol{w}_{i-1})-\nabla J_k(\boldsymbol{w}_{i-1})\right\|^2|\boldsymbol{w}_{i-1}\right\}+2\delta_k^2\|w_k^o-\boldsymbol{w}_{i-1}\|^2+2\delta^2\|w^o-\boldsymbol{w}_{i-1}\|^2\right)\\
&\overset{(4.27)}{\leq}\frac{1}{K}\sum_{k=1}^{K}\left(\left(\beta_k^2+2\delta_k^2\right)\|w_k^o-\boldsymbol{w}_{i-1}\|^2+\sigma_k^2+2\delta^2\|w^o-\boldsymbol{w}_{i-1}\|^2\right)\\
&\overset{(b)}{\leq}\frac{1}{K}\sum_{k=1}^{K}\left(\left(2\beta_k^2+4\delta_k^2+2\delta^2\right)\|w^o-\boldsymbol{w}_{i-1}\|^2+2\left(\beta_k^2+2\delta_k^2\right)\|w_k^o-w^o\|^2+\sigma_k^2\right)\\
&\overset{(c)}{=}\beta^2\|w^o-\boldsymbol{w}_{i-1}\|^2+\sigma^2
\end{aligned}
\tag{4.54}
$$

where (a) follows from smoothess conditions on $J(w)$ and $J_k(w)$ and (b) follows from $\|a+b\|^2\leq 2\|a\|^2+2\|b\|^2$. In (c) we defined:

$$\beta^2=\frac{1}{K}\sum_{k=1}^{K}\left(2\beta_k^2+4\delta_k^2+2\delta^2\right) \tag{4.55}$$

$$\sigma^2=\frac{1}{K}\sum_{k=1}^{K}\left(2\left(\beta_k^2+2\delta_k^2\right)\|w_k^o-w^o\|^2+\sigma_k^2\right) \tag{4.56}$$

From Theorem 3.1, we can then conclude:

$$\lim_{i\to\infty}\mathbb{E}\|w^o-\boldsymbol{w}_i\|^2\leq\frac{\mu}{\nu K}\sum_{k=1}^{K}\left(2\left(\beta_k^2+2\delta_k^2\right)\|w_k^o-w^o\|^2+\sigma_k^2\right) \tag{4.57}$$

In the homogenous case where $w_k^o=w^o$, $\sigma_k^2=\sigma^2$, this simplifies to:

$$\lim_{i\to\infty}\mathbb{E}\|w^o-\boldsymbol{w}_i\|^2\leq\frac{\mu\sigma^2}{\nu} \tag{4.58}$$

which is the same performance as a non-cooperative approach. This is to be expected, as only a single agent is participating in the learning protocol at any given iteration.

4.6 PROBLEMS

4.1 In this problem we will demonstrate the linear performance gain promised by relation (4.46) in code for Example 4.2. Formulate a model w^o of your choice, and generate data according to the observation model (4.39), ensuring agents are homogenous by sampling $\boldsymbol{h}_k$ and $\boldsymbol{v}_k$ from identical distributions. Reasonable choices are $\boldsymbol{h}_k \sim \mathcal{N}(0, \sigma_h^2 I_M)$ and $\boldsymbol{v}_k \sim \mathcal{N}(0, \sigma_v^2)$. Implement recursions (4.42)–(4.43) and plot the evolution of the error over time. Compare the performance for $K = 1$, $K = 10$ and $K = 100$ agents and verify whether you observe linear gains in performance.

4.2 Consider local empirical risk minimization problems with unbalanced datasets of the form:

$$J_k(w) = \frac{1}{N_k} \sum_{n=1}^{N_k} Q(w; x_{k,n}) \tag{4.59}$$

and the global empirical risk minimization problem:

$$J(w) = \frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N_k} Q(w; x_{k,n}) \tag{4.60}$$

where we defined $N = \sum_{k=1}^{K} N_k$. Show that $J(w)$ can be written as a *weighted* consensus problem of the form:

$$J(w) = \sum_{k=1}^{K} p_k J_k(w) \tag{4.61}$$

where p_k are suitably chosen weights satisfying $\sum_{k=1}^{K} p_k = 1$.