Multi-Agent Optimization and Learning
Lecture 1: Foundations

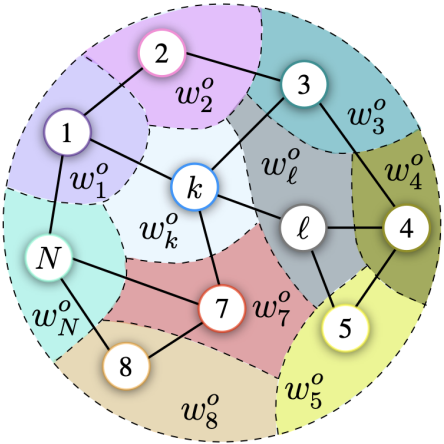
Stefan Vlaski[†] and Ali H. Sayed[⋆]

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

IEEE ICASSP 2024 Short Course

Multi-Agent Systems with Dispersed Data and Capabilities

- Sensor networks
- Autonomous vehicles
- Social networks
- Power grids
- Mobile devices
- Drone swarms
- Many more



Objectives of this Course

Objective

To provide attendees with a fundamental understanding and the resulting intuition about distributed algorithms and the resulting performance trade-offs in intelligent multi-agent systems.

Hopefully, by the end of this course:

- **Those interested in developing distributed algorithms** will have the tools needed to rigorously develop and analyze new algorithms for distributed learning.
- **Those interested in applying distributed algorithms** will have an understanding of fundamental properties that allow for informed decisions on when and how to use and adapt distributed algorithms in their domain.

Structure of this course

- **Lecture 1:** Foundations
  - Taxonomy, inference, learning and optimization, (stochastic) gradient descent.
- **Lecture 2:** Learning with a fusion center
  - Partial participation, local updates, heterogeneity and their impact on performance.
- **Lecture 3:** Graphs and their role decentralized processing
  - Graph spectral theory, averaging over graphs.
- **Lecture 4:** Algorithms for decentralized optimization and learning
  - Penalty-based (Distributed gradient descent and diffusion), primal-dual (EXTRA and Exact diffusion), and gradient-tracking (DIGing, NEXT and AugDGM)
- **Lecture 5:** Performance guarantees and trade-offs
  - Effect of heterogeneity and bias-correction, graph connectivity, gradient noise.
- **Lecture 6:** Advanced topics and open problems
  - Multi-task and meta-learning, compression, privacy.

Outline of This Lecture

In this lecture we introduce some foundational concepts in a manner that will generalize and translate to distributed structures in the rest of the course.

- Components of a multi-agent system, and a taxonomy of distributed algorithms
- Inference, Learning and Optimization
- Deterministic and Stochastic Gradient Algorithms
- Centralized Learning

Components of a Multi-Agent System

- **Agents:** Individual entity with basic capabilities to observe data, perform computations and communicate. Depending on the application, the term "agent" can represent a multitude of things, ranging from a sensor, to mobile devices such as phones, autonomous vehicles and drones, or humans.
- **Data:** Agents will be equipped with the ability to observe the world around them, and these observations will be captured in the form of data. These data may be available as a batch of fixed-size, or be observed on the fly. We will model data as random variables.
- **Objective:** Agents will aim to achieve some objective, which will be represented as an optimization problem.
- **Model:** In most applications, agents will maintain and refine a model, captured through parameters, which reconciles observed data and the objective. Models will be learned by adjusting the parameterization to optimize the agent's objective.
- **Network:** We will represent communication links between agents through a network topology, which will place limits on the flow of information. We elaborate on some typical structures and taxonomy in the next slides.

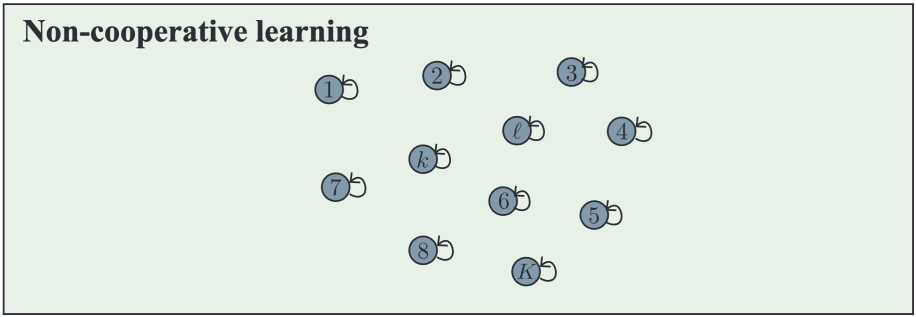


Figure: Non-cooperative approaches serve as a worst-case baseline.

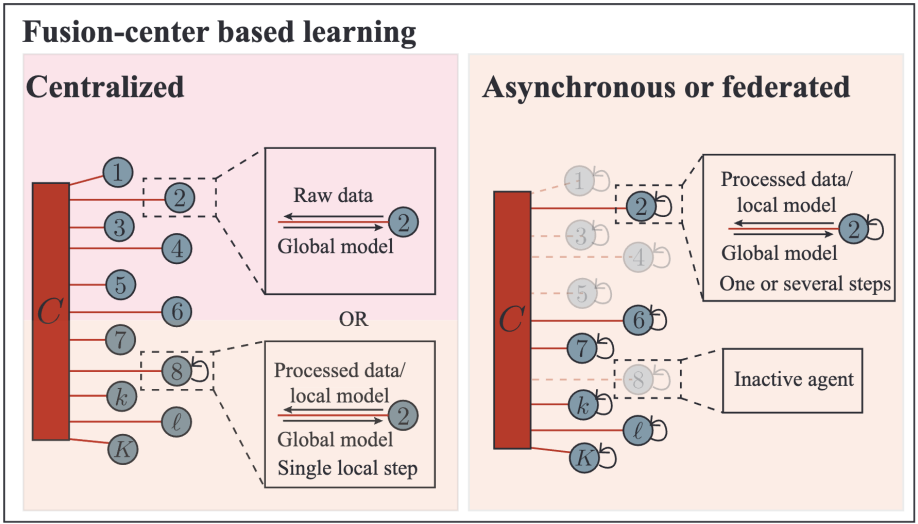


Figure: Centralized structures serve as a best-case baseline.

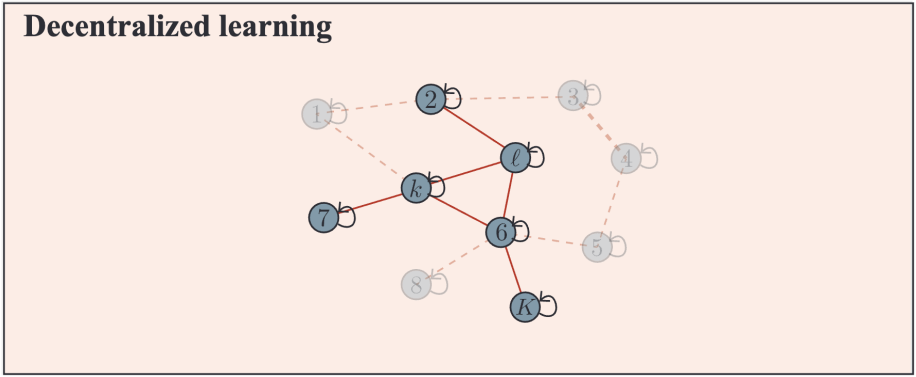


Figure: Decentralized structures avoid a fusion center.

Maximum A Posteriori Estimation

If we are provided with the conditional distribution $f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h)$ along with a single realization of the feature $\boldsymbol{h} \in \mathbb{R}^{M_h}$, it is quite natural to estimate $\boldsymbol{\gamma} \in \mathbb{R}^{M_\gamma}$ as the most likely outcome given $\boldsymbol{h}$. We can express this formally as:

$$\gamma^\star \triangleq \arg\max_{\gamma \in \Gamma} f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h) \tag{1}$$

The conditional distribution $f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h)$ is frequently referred to as the *posterior distribution* of $\boldsymbol{\gamma}$, making $\gamma^\star$ the *maximum a posteriori (MAP) estimate* of $\boldsymbol{\gamma}$ given $\boldsymbol{h}$. We can also regard the MAP solution as a random variable, denoted in boldface notation $\boldsymbol{\gamma}^\star$, when it is viewed as a function of the random observation $\boldsymbol{h}$:

$$\boldsymbol{\gamma}^\star \triangleq \arg\max_{\gamma \in \Gamma} f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|\boldsymbol{h}) \tag{2}$$

The Bayesian Learning Framework

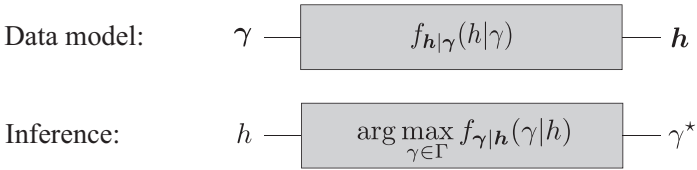


Figure: A general model underlying most inference problems. The top row illustrates the generative model for the observations $\boldsymbol{h}$, while the bottom row illustrates the Bayesian inference formulation for recovering the label variable.

1

[1] We use **bold** font to describe a random variable, while regular font denotes its realization.

Example: Linear Regression

Let us consider a feature $\boldsymbol{h} \in \mathbb{R}^M$, and a linear model parameterized by $w^o \in \mathbb{R}^M$, resulting in the label $\boldsymbol{\gamma} \in \mathbb{R}$:

$$\boldsymbol{\gamma} = \boldsymbol{h}^{\mathsf{T}} w^o + \boldsymbol{v} \tag{3}$$

Here, $\boldsymbol{v}$ denotes zero-mean noise, which we assume to be normally distributed $\boldsymbol{v} \sim \mathcal{N}(0, \sigma_v^2)$ with variance $\sigma_v^2 > 0$. Suppose we would like to predict the most likely label $\boldsymbol{\gamma}$ for a given feature $\boldsymbol{h}$. We can first evaluate the conditional pdf:

$$f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h) = \frac{1}{\sqrt{2\pi\sigma_v^2}} e^{-\frac{1}{2\sigma_v^2}(\gamma - h^{\mathsf{T}} w^o)^2} \tag{4}$$

Example: Linear Regression

We see directly that the posterior distribution $f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h)$ is maximized whenever $(\gamma - h^{\mathsf{T}}w^o)^2 = 0$ and hence $\gamma^\star = h^{\mathsf{T}}w^o$. An alternative approach is to take the logarithm of the posterior and find:

$$\log f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h) = \log\left(\frac{1}{\sqrt{2\pi\sigma_v^2}}\right) - \frac{1}{2\sigma_v^2}(\gamma - h^{\mathsf{T}}w^o)^2 \tag{5}$$

Since the logarithm of a positive variable is strictly increasing, it follows that the maximizing argument of $f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h)$ is the same as the maximizing argument of $\log f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h)$, which is the choice γ that satisfies $\frac{1}{2\sigma_v^2}(\gamma - h^{\mathsf{T}}w^o)^2 = 0$. It then follows that:

$$\gamma^\star = \arg\max_{\gamma\in\Gamma} f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h) = h^{\mathsf{T}}w^o \tag{6}$$

Limitations

In its current form, however, the framework has two limitations.

- First, relation (1) and the following examples provided a rule to map a single observation h to an estimate $\gamma^\star$ of γ. In many inference and learning settings, we will be provided with multiple realizations $\{h_n\}_{n=1}^N$ of the random variable $\boldsymbol{h}$, and seek to use these aggregate realizations in generating $\gamma^\star$.
- Perhaps more importantly, the resulting estimation rules are driven in large part by the conditional probability densities (e.g., $f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h)$ when estimating $\boldsymbol{\gamma}$ from $\boldsymbol{h}$). This essentially requires full knowledge of the statistical model relating the label $\boldsymbol{\gamma}$ to its feature $\boldsymbol{h}$. For instance, in the case of the linear channel treated in the previous example, this translates into knowledge of the channel taps contained in w^o, along with the distribution of the noise term $\boldsymbol{v}$. In practice, we do not expect to have access to so much knowledge, and need to estimate relevant conditional distributions directly from data. We refer to the process of estimating models from data as "learning" and develop it from a Bayesian perspective.

Multiple realizations

Dealing with multiple realizations in straightforward for i.i.d. data:

$$f_{\{\boldsymbol{h}_n\}_{n=1}^N|\boldsymbol{\gamma}}\left(\{h_n\}_{n=1}^N|\gamma\right) \overset{(a)}{=} \prod_{n=1}^N f_{\boldsymbol{h}_n|\boldsymbol{\gamma}}\left(h_n|\gamma\right) \overset{(b)}{=} \prod_{n=1}^N f_{\boldsymbol{h}|\boldsymbol{\gamma}}\left(h_n|\gamma\right) \tag{7}$$

where (a) holds by conditional independence and (b) holds by identical distribution of the random variables $\boldsymbol{h}_n$. We can then adjust the framework (1) to include multiple observations as follows:

$$\begin{aligned}\gamma^\star \triangleq \arg\max_{\gamma\in\Gamma} f_{\boldsymbol{\gamma}|\{\boldsymbol{h}_n\}_{n=1}^N}\left(\gamma|\,\{h_n\}_{n=1}^N\right) &= \arg\max_{\gamma\in\Gamma} \frac{f_{\{\boldsymbol{h}_n\}_{n=1}^N|\boldsymbol{\gamma}}\left(\{h_n\}_{n=1}^N|\gamma\right)\times f_{\boldsymbol{\gamma}}(\gamma)}{f_{\{\boldsymbol{h}_n\}_{n=1}^N}\left(\{h_n\}_{n=1}^N\right)} \\ &= \arg\max_{\gamma\in\Gamma} f_{\{\boldsymbol{h}_n\}_{n=1}^N|\boldsymbol{\gamma}}\left(\{h_n\}_{n=1}^N|\gamma\right)\times f_{\boldsymbol{\gamma}}(\gamma) \\ &= \arg\max_{\gamma\in\Gamma}\left(\prod_{n=1}^N f_{\boldsymbol{h}|\boldsymbol{\gamma}}\left(h_n|\gamma\right)\right)\times f_{\boldsymbol{\gamma}}(\gamma)\end{aligned} \tag{8}$$

Multiple realizations

The fact that the logarithmic function is monotonically increasing ensures that:

$$\begin{aligned}\gamma^\star &= \arg\max_{\gamma\in\Gamma} \log\left[\left(\prod_{n=1}^{N} f_{\boldsymbol{h}|\boldsymbol{\gamma}}\left(h_n|\gamma\right)\right) \times f_{\boldsymbol{\gamma}}(\gamma)\right] \\ &= \arg\max_{\gamma\in\Gamma}\left\{\sum_{n=1}^{N} \log f_{\boldsymbol{h}|\boldsymbol{\gamma}}\left(h_n|\gamma\right) + \log f_{\boldsymbol{\gamma}}(\gamma)\right\} \\ &= \arg\min_{\gamma\in\Gamma}\left\{-\sum_{n=1}^{N} \log f_{\boldsymbol{h}|\boldsymbol{\gamma}}\left(h_n|\gamma\right) - \log f_{\boldsymbol{\gamma}}(\gamma)\right\} \end{aligned} \tag{9}$$

We can normalize by the sample size N, since the minimizing argument $\gamma^\star$ is unaffected by scaling:

$$\gamma^\star = \arg\min_{\gamma\in\Gamma}\left\{-\frac{1}{N}\sum_{n=1}^{N} \log f_{\boldsymbol{h}|\boldsymbol{\gamma}}\left(h_n|\gamma\right) - \frac{1}{N}\log f_{\boldsymbol{\gamma}}(\gamma)\right\} \tag{10}$$

Example: Inverting a linear regression model

We consider a binary class variable $\boldsymbol{\gamma} \in \{+1, -1\}$. Suppose that the observed feature $\boldsymbol{h}$ is normally distributed with mean that depends on the class variable $\boldsymbol{\gamma}$. Specifically, we model $\boldsymbol{h} \sim \mathcal{N}\left(w^o, \sigma_v^2 I_M\right)$ when $\gamma = +1$ and $\boldsymbol{h} \sim \mathcal{N}\left(-w^o, \sigma_v^2 I_M\right)$ when $\gamma = -1$. This relation can be represented through the linear model:

$$\boldsymbol{h} = \boldsymbol{\gamma} w^o + \boldsymbol{v} \tag{11}$$

with $\boldsymbol{v} \sim \mathcal{N}(0, \sigma_v^2)$. In this sense, we may view the present example as the inverse problem to the linear regression example we previously saw. The linear relationship translates into the conditional distribution:

$$f_{\boldsymbol{h}|\boldsymbol{\gamma}}(h|\gamma) = \frac{1}{\sqrt{(2\pi)^M \sigma_v^{2M}}} e^{-\frac{1}{2\sigma_v^2}\|h - \gamma w^o\|^2} \tag{12}$$

Example: Inverting a linear regression model

Applying (10), and assuming an equal prior $\mathbb{P}\{\gamma = +1\} = \mathbb{P}\{\gamma = -1\} = \frac{1}{2}$, we obtain:

$$\begin{aligned}
\gamma^\star &= \arg\min_{\gamma \in \pm 1} \frac{1}{N} \sum_{n=1}^{N} \|h_n - \gamma w^o\|^2 \\
&= \arg\min_{\gamma \in \pm 1} \left\{ \frac{1}{N} \sum_{n=1}^{N} \|h_n\|^2 - 2\gamma \left(\frac{1}{N} \sum_{n=1}^{N} h_n \right)^{\mathsf{T}} w^o + \|\gamma w^o\|^2 \right\} \\
&\overset{(a)}{=} \arg\min_{\gamma \in \pm 1} \left\{ -2\gamma \left(\frac{1}{N} \sum_{n=1}^{N} h_n \right)^{\mathsf{T}} w^o + \gamma^2 \|w^o\|^2 \right\} \\
&\overset{(b)}{=} \arg\min_{\gamma \in \pm 1} \left\{ -2\gamma \left(\frac{1}{N} \sum_{n=1}^{N} h_n \right)^{\mathsf{T}} w^o \right\} = \operatorname{sign} \left\{ \left(\frac{1}{N} \sum_{n=1}^{N} h_n \right)^{\mathsf{T}} w^o \right\}
\end{aligned} \tag{13}$$

In this sequence of reformulations, (a) follows from the fact that $\frac{1}{N} \sum_{n=1}^{N} \|h_n\|^2$ is independent of γ, (b) uses the fact that $\boldsymbol{\gamma}^2 = 1$.

From Inference to Learning

So far, given a known model $f_{\boldsymbol{h}|\boldsymbol{\gamma}}$ and realizations of $\boldsymbol{h}$, we performed *inference* of the label $\boldsymbol{\gamma}$. To formalized the *learning* process, we will assume that the conditional likelihood $f_{\boldsymbol{h}|\boldsymbol{\gamma}}(\cdot|\gamma)$ is parameterized by a learnable parameter $w \in \mathbb{R}^{M_w}$ and write instead $f_{\boldsymbol{h}|\boldsymbol{\gamma},\boldsymbol{w}}(\cdot|\gamma, w)$. Under this parameterization, learning the conditional distribution of $\boldsymbol{\gamma}$ given $\boldsymbol{h}$ is equivalent to learning the parameter $\boldsymbol{w}$ that parameterizes $f_{\boldsymbol{h}|\boldsymbol{\gamma},\boldsymbol{w}}(\cdot|\gamma, w)$.

To formulate a procedure for learning $\boldsymbol{w}$ from pairs $\{\boldsymbol{h}, \boldsymbol{\gamma}\}$ we mirror the argument so far. Suppose the model $\boldsymbol{w}$ is sampled once, yielding the realization w^o. The *training data* $\{\boldsymbol{h}_n, \boldsymbol{\gamma}_n\}_{n=1}^N$ is sampled N times, where each pair $\{\boldsymbol{h}_n, \boldsymbol{\gamma}_n\}$ is sampled from $f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}(h, \gamma|w^o)$. If we suppose that feature-label pairs $\{\boldsymbol{h}_n, \boldsymbol{\gamma}_n\}$ are identically and independently distributed after conditioning on the parameterization $\boldsymbol{w}$, we can factorize:

$$
\begin{aligned}
f_{\{\boldsymbol{h}_n,\boldsymbol{\gamma}_n\}_{n=1}^N|\boldsymbol{w}}\left(\{h_n, \gamma_n\}_{n=1}^N|w\right) &\overset{(a)}{=} \prod_{n=1}^N f_{\boldsymbol{h}_n,\boldsymbol{\gamma}_n|\boldsymbol{w}}(h_n, \gamma_n|w) \\
&\overset{(b)}{=} \prod_{n=1}^N f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}(h_n, \gamma_n|w) \qquad (14)
\end{aligned}
$$

From Inference to Learning

We can then define the MAP estimate of the weight vector w as:

$$
\begin{aligned}
w^\star &\triangleq \arg\max_{w\in\mathbb{R}^{M_w}} f_{\boldsymbol{w}|\{\boldsymbol{h}_n,\boldsymbol{\gamma}_n\}_{n=1}^N}\left(w|\,\{h_n,\gamma_n\}_{n=1}^N\right) \\
&= \arg\max_{w\in\mathbb{R}^{M_w}} \frac{f_{\{\boldsymbol{h}_n,\boldsymbol{\gamma}_n\}_{n=1}^N|\boldsymbol{w}}\left(\{h_n,\gamma_n\}_{n=1}^N|w\right)\times f_{\boldsymbol{w}}(w)}{f_{\{\boldsymbol{h}_n,\boldsymbol{\gamma}_n\}_{n=1}^N}\left(\{h_n,\gamma_n\}_{n=1}^N\right)} \\
&= \arg\max_{w\in\mathbb{R}^{M_w}} f_{\{\boldsymbol{h}_n,\boldsymbol{\gamma}_n\}_{n=1}^N|\boldsymbol{w}}\left(\{h_n,\gamma_n\}_{n=1}^N|w\right)\times f_{\boldsymbol{w}}(w) \\
&= \arg\max_{w\in\mathbb{R}^{M_w}} \left(\prod_{n=1}^N f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}\left(h_n,\gamma_n|w\right)\right)\times f_{\boldsymbol{w}}(w) \qquad (15)
\end{aligned}
$$

Following the same argument that led to (10), we arrive at:

$$
w^\star = \arg\min_{w\in\mathbb{R}^M}\left\{-\frac{1}{N}\sum_{n=1}^N \log f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}\left(h_n,\gamma_n|w\right)-\frac{1}{N}\log f_{\boldsymbol{w}}(w)\right\} \qquad (16)
$$

Distinction between the true model and the MAP estimate

Observe that we make a deliberate distinction between the *true* model w^o, which parameterizes the distribution $f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}(h,\gamma|w^o)$ from which the samples $\{\boldsymbol{h}_n,\boldsymbol{\gamma}_n\}_{n=1}^N$ are generated, and the MAP *estimate* $w^\star$, which maximizes the posterior distribution of $\boldsymbol{w}$ after observing the samples $\{\boldsymbol{h}_n,\boldsymbol{\gamma}_n\}_{n=1}^N$. In general, there will be a difference between the MAP model $w^\star$ and the true model w^o. The expectation is that as the size of the data set N grows, the MAP model $w^\star$ will become increasingly accurate and approach w^o as $N \to \infty$. Detailed proofs of generalization are beyond the scope of this course.

Train and Test Data

Once we have found a $w^\star$ based on some training set $\{h_n, \gamma_n\}_{n=1}^N$, we can then apply the model to an unseen feature to perform *approximate* Bayesian inference:

$$\gamma^{\text{test}\star} \triangleq \arg\max_{\gamma \in \Gamma} f_{\boldsymbol{\gamma}|\boldsymbol{h},\boldsymbol{w}}(\gamma | h^{\text{test}}, w^\star) \tag{17}$$

We emphasize that (17) is only an *approximate* MAP estimate for the true label γ^{test}, since it is generated using an estimate of the posterior distribution $f_{\boldsymbol{\gamma}|\boldsymbol{h},\boldsymbol{w}}(\gamma | h^{\text{test}}, w^\star)$ using the learned parameterization $w^\star$. In general, and in particular for finite sample sizes N of the training data, the learned parameterization $w^\star$ will be different from the true parameterization w^o that actually generated the data. The difference between predictions made using the true model w^o and the learned model $w^\star$ is known as a *generalization error*.

Compact notation for feature-label pairs

As is evident from (16), during learning, we are provided with feature-label pairs $\{\boldsymbol{h}, \boldsymbol{\gamma}\}$ or their realizations $\{h_n, \gamma_n\}_{n=1}^N$. To simplify the notation, we will collect features $\boldsymbol{h} \in \mathbb{R}^{M_{\boldsymbol{h}}}$ and labels $\boldsymbol{\gamma} \in \mathbb{R}^{M_{\boldsymbol{\gamma}}}$ into a single augmented data vector $\boldsymbol{x} \in \mathbb{R}^{M_{\boldsymbol{h}}+M_{\boldsymbol{\gamma}}}$, such that:

$$\boldsymbol{x} \triangleq \operatorname{col}\{\boldsymbol{h}, \boldsymbol{\gamma}\} = \begin{pmatrix} \boldsymbol{h} \\ \boldsymbol{\gamma} \end{pmatrix} \tag{18}$$

Similarly, we will collect realizations into $x_n \triangleq \operatorname{col}\{h_n, \gamma_n\}$. In this manner, we can write more compactly:

$$f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}(h, \gamma|w) = f_{\boldsymbol{x}|\boldsymbol{w}}(x|w) \tag{19}$$

The MAP learning problem (16) then becomes:

$$w^\star = \arg\min_{w \in \mathbb{R}^M} \left\{ -\frac{1}{N} \sum_{n=1}^{N} \log f_{\boldsymbol{x}|\boldsymbol{w}}(x_n|w) - \frac{1}{N} \log f_{\boldsymbol{w}}(w) \right\} \tag{20}$$

Ridge regression for linear models

Let us consider again the linear model from previous examples.

$$\boldsymbol{\gamma} = \boldsymbol{h}^{\mathsf{T}} w^o + \boldsymbol{v} \tag{21}$$

We can interpret w^o as the realization of a random variable $\boldsymbol{w}$, and assume that the parameter $\boldsymbol{w}$ is independent of all other random variables. Suppose we impose a normal prior of the form $\boldsymbol{w} \sim \mathcal{N}(0, \sigma_w^2 I_M)$. We continue to assume $\boldsymbol{v} \sim \mathcal{N}(0, \sigma_v^2)$ and $\boldsymbol{h} \sim \mathcal{N}(0, \sigma_h^2 I_M)$. It then follows that:

$$f_{\boldsymbol{w}}(w) = \frac{1}{\sqrt{(2\pi)^M \sigma_w^{2M}}} e^{-\frac{1}{2\sigma_w^2}\|w\|^2} \tag{22}$$

$$\begin{aligned} f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}(h,\gamma|w) &= f_{\boldsymbol{\gamma}|\boldsymbol{h},\boldsymbol{w}}(\gamma|h,w) \times f_{\boldsymbol{h}|\boldsymbol{w}}(h|w) \\ &\overset{(a)}{=} f_{\boldsymbol{\gamma}|\boldsymbol{h},\boldsymbol{w}}(\gamma|h,w) \times f_{\boldsymbol{h}}(h) \\ &= \frac{1}{\sqrt{2\pi\sigma_v^2}} e^{-\frac{1}{2\sigma_v^2}(\gamma - h^{\mathsf{T}} w)^2} \times \frac{1}{\sqrt{(2\pi)^M \sigma_h^{2M}}} e^{-\frac{1}{2\sigma_h^2}\|h\|^2} \end{aligned} \tag{23}$$

Ridge regression for linear models

Substituting these relations into (16), and removing constant terms independent of w, we obtain:

$$w^\star = \arg\min_{w\in\mathbb{R}^M} \left\{ \frac{1}{2N}\sum_{n=1}^{N}\left(\gamma_n - h_n^\mathsf{T} w\right)^2 + \frac{1}{2N}\frac{\sigma_v^2}{\sigma_w^2}\|w\|^2 \right\} \tag{24}$$

which corresponds to a least-squares regression problem, where the fit of the parameter w is evaluated by measuring the squared error of the prediction $h_n^\mathsf{T} w$ to the observation γ_n. In practice, the variances σ_v^2 and σ_w^2 are unknown, and it is common to instead introduce a *hyperparameter* ρ resulting in:

$$w^\star = \arg\min_{w\in\mathbb{R}^M} \left\{ \frac{1}{N}\sum_{n=1}^{N}\left(\gamma_n - h_n^\mathsf{T} w\right)^2 + \rho\|w\|^2 \right\} \tag{25}$$

The hyperparameter ρ is either fixed a priori based on an estimate of the ratio $\frac{\sigma_v^2}{\sigma_w^2}$, or tuned on a subset of the training data using cross-validation.

Asymptotic Behavior

It is instructive to examine the asymptotic behavior of the optimal estimate (16) as the sample size N grows. We have for all w such that $f_{\boldsymbol{w}}(w) > 0$:

$$\begin{aligned}
&\lim_{N\to\infty} \left\{ -\frac{1}{N} \sum_{n=1}^{N} \log f_{\boldsymbol{x}\,|\boldsymbol{w}}\,(x_n|w) - \frac{1}{N} \log f_{\boldsymbol{w}}(w) \right\} \\
&= \lim_{N\to\infty} \left\{ -\frac{1}{N} \sum_{n=1}^{N} \log f_{\boldsymbol{x}\,|\boldsymbol{w}}\,(x_n|w) \right\} - \lim_{N\to\infty} \left\{ \frac{1}{N} \log f_{\boldsymbol{w}}(w) \right\} \\
&\overset{(a)}{=} \lim_{N\to\infty} \left\{ -\frac{1}{N} \sum_{n=1}^{N} \log f_{\boldsymbol{x}\,|\boldsymbol{w}}\,(x_n|w) \right\} \\
&\overset{(b)}{=} -\,\mathbb{E}_{\boldsymbol{x}\sim f_{\boldsymbol{x}\,|\,\boldsymbol{w}}(x|w^o)} \Big\{ \log f_{\boldsymbol{x}\,|\boldsymbol{w}}\,(\boldsymbol{x}\,|w) \Big\}
\end{aligned} \tag{26}$$

Here, (a) follows since $f_{\boldsymbol{w}}(w) > 0 \Longrightarrow |\log f_{\boldsymbol{w}}(w)| < \infty$ and hence:

$$\lim_{N\to\infty} \left\{ \frac{1}{N} \log f_{\boldsymbol{w}}(w) \right\} = 0 \tag{27}$$

Example: Mean-Square-Error (MSE) estimation

Considering the linear model from previous examples, we find from (23):

$$
\begin{aligned}
& -\log f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}\,(h,\gamma|w) \\
\overset{(23)}{=} & -\log\left(\frac{1}{\sqrt{2\pi\sigma_v^2}}e^{-\frac{1}{2\sigma_v^2}(\gamma-h^{\mathsf{T}}w)^2}\times\frac{1}{\sqrt{(2\pi)^M\sigma_h^{2M}}}e^{-\frac{1}{2\sigma_h^2}\|h\|^2}\right) \\
= & -\log\left(e^{-\frac{1}{2\sigma_v^2}(\gamma-h^{\mathsf{T}}w)^2}\right)\underbrace{-\log\left(\frac{1}{\sqrt{2\pi\sigma_v^2}}\times\frac{1}{\sqrt{(2\pi)^M\sigma_h^{2M}}}e^{-\frac{1}{2\sigma_h^2}\|h\|^2}\right)}_{\triangleq C} \\
= & \frac{1}{2\sigma_v^2}\Big(\gamma-h^{\mathsf{T}}w\Big)^2+C
\end{aligned}
\tag{28}
$$

Since $\frac{1}{\sigma_v^2}$ and C are independent of w, we conclude from (26) after shifting and scaling:

$$
w^o=\arg\min_w\frac{1}{2}\mathbb{E}\Big(\boldsymbol{\gamma}-\boldsymbol{h}^{\mathsf{T}}w\Big)^2 \tag{29}
$$

Example: Mean-Square-Error (MSE) estimation

Since the risk function in this case is quadratic in w, we can obtain a closed-form expression for the minimizer w^o. Indeed, after differentiating, we find:

$$\nabla\left(\frac{1}{2}\mathbb{E}\Big(\boldsymbol{\gamma}-\boldsymbol{h}^{\mathsf{T}}w\Big)^2\right)=-\underbrace{\mathbb{E}\boldsymbol{\gamma}\boldsymbol{h}}_{\triangleq r_{\gamma h}}+\underbrace{\mathbb{E}\boldsymbol{h}\boldsymbol{h}^{\mathsf{T}}}_{\triangleq R_h}w=-r_{\gamma h}+R_h w \tag{30}$$

At w^o, this derivative must be equal to zero, and hence after rearranging:

$$w^o=R_h^{-1}r_{\gamma h} \tag{31}$$

We can verify that the minimizing argument w^o of the MSE risk (29) indeed corresponds to the *true* model parameters in (3). To this end, note from

$$\boldsymbol{\gamma}=\boldsymbol{h}^{\mathsf{T}}w^o+\boldsymbol{v} \tag{32}$$

that if we multiply by $\boldsymbol{h}$ from the left and take expectations, we get:

$$r_{\gamma h}=R_h w^o \tag{33}$$

Empirical and expected risk minimization for single agent learning

- We have hence motivated the following optimization problems as rigorous learning frameworks.
- General empirical risk minimization problems take the form:

$$w^\star = \arg\min_w \frac{1}{N}\sum_{n=1}^{N} Q(w; x_n) + R(w) \tag{34}$$

  Here, $Q(w; x_n)$ denotes the *loss* of the parameterization w on the sample x_n. The term $R(w)$ corresponds to a regularization term, which is independent of the data x_n, but may depend on the sample size N.
- (Expected) risk minimization problems take the form:

$$w^o = \arg\min_w \mathbb{E}_{\boldsymbol{x}} Q(w; \boldsymbol{x}) \tag{35}$$

Extension to Multi-Agent Systems

To this end, we consider a collection of K agents. Each agent has sensing capabilities, and collects a local dataset $\{\boldsymbol{h}_{k,n}, \boldsymbol{\gamma}_{k,n}\}_{n=1}^{N}$. For simplicity, we assume that the sample size N is common to all agents. Each pair of samples is sampled independently and identically from the distribution:

$$f_{\boldsymbol{h}_{k,n},\boldsymbol{\gamma}_{k,n}|\boldsymbol{w}_k}\left(h,\gamma|w^o\right) = f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}\left(h,\gamma|w^o\right) \tag{36}$$

Now assume first that the different agents are disconnected, with no way to exchange information among them. In that case, it is reasonable to formulate independent local learning problems:

$$w_k^\star \triangleq \arg\max_{w\in\mathbb{R}^M} f_{\boldsymbol{w}|\{\boldsymbol{h}_{k,n},\boldsymbol{\gamma}_{k,n}\}_{n=1}^{N}}\left(w|\,\{h_{k,n},\gamma_{k,n}\}_{n=1}^{N}\right) \tag{37}$$

Extension to Multi-Agent Systems

Following the argument same argument as before, we arrive at the equivalent formulation:

$$\begin{aligned} w_k^\star &= \arg\min_{w\in\mathbb{R}^M} \left\{ -\frac{1}{N}\sum_{n=1}^{N} \log f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}\left(h_{k,n},\gamma_{k,n}|w\right) - \frac{1}{N}\log f_{\boldsymbol{w}}(w) \right\} \\ &= \arg\min_{w\in\mathbb{R}^M} J_k(w) \end{aligned} \tag{38}$$

where we defined:

$$J_k(w) \triangleq -\frac{1}{N}\sum_{n=1}^{N} \log f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}\left(h_{k,n},\gamma_{k,n}|w\right) - \frac{1}{N}\log f_{\boldsymbol{w}}(w) \tag{39}$$

Through the agent subscript k we emphasize here that because the local samples $\{\boldsymbol{h}_{k,n},\boldsymbol{\gamma}_{k,n}\}_{n=1}^N$ are distinct, this implies that $J_k(\cdot)$ as well as the resulting optimal models $w_k^\star$ are distinct as well. While we can claim local optimality of each model $w_k^\star$ in the sense of (38), this model is making use only of the data available to agent k.

Extension to Multi-Agent Systems

As an alternative to (38), we can instead consider the *global* learning problem:

$$w^\star \triangleq \arg\max_{w\in\mathbb{R}^M} f_{\boldsymbol{w}|\left\{\{\boldsymbol{h}_{k,n},\boldsymbol{\gamma}_{k,n}\}_{n=1}^N\right\}_{k=1}^K}\left(w|\left\{\{h_{k,n},\gamma_{k,n}\}_{n=1}^N\right\}_{k=1}^K\right) \tag{40}$$

As long as the samples $\{\boldsymbol{h}_{k,n},\boldsymbol{\gamma}_{k,n}\}_{n=1}^N$ are also mutually independent between agents after conditioning on $\boldsymbol{w}$, we can factorize the conditional likelihoods:

$$\begin{aligned} w^\star &\triangleq \arg\max_{w\in\mathbb{R}^M} f_{\boldsymbol{w}|\left\{\{\boldsymbol{h}_{k,n},\boldsymbol{\gamma}_{k,n}\}_{n=1}^N\right\}_{k=1}^K}\left(w|\left\{\{h_{k,n},\gamma_{k,n}\}_{n=1}^N\right\}_{k=1}^K\right) \\ &= \arg\max_{w\in\mathbb{R}^M} f_{\left\{\{\boldsymbol{h}_{k,n},\boldsymbol{\gamma}_{k,n}\}_{n=1}^N\right\}_{k=1}^K|\boldsymbol{w}}\left(\left\{\{h_{k,n},\gamma_{k,n}\}_{n=1}^N\right\}_{k=1}^K|w\right)\times f_{\boldsymbol{w}}(w) \\ &= \arg\max_{w\in\mathbb{R}^M}\left(\prod_{k=1}^K f_{\left\{\boldsymbol{h}_{k,n},\boldsymbol{\gamma}_{k,n}\right\}_{n=1}^N|\boldsymbol{w}}\left(\{h_{k,n},\gamma_{k,n}\}_{n=1}^N|w\right)\right)\times f_{\boldsymbol{w}}(w) \end{aligned} \tag{41}$$

Aggregate Optimization Problem

After taking logarithms, and normalizing, we find:

$$\begin{aligned} w^\star &= \arg\min_w \left\{ -\frac{1}{KN} \sum_{k=1}^{K} \sum_{n=1}^{N} \log f_{\boldsymbol{h},\boldsymbol{\gamma}|\boldsymbol{w}}\left(h_{k,n}, \gamma_{k,n}|w\right) - \frac{1}{KN} \log f_{\boldsymbol{w}}(w) \right\} \\ &= \arg\min_w \frac{1}{K} \sum_{k=1}^{K} J_k(w) \\ &= J(w) \end{aligned} \tag{42}$$

where we defined:

$$\boxed{J(w) = \frac{1}{K} \sum_{k=1}^{K} J_k(w)} \tag{43}$$

This problem is known as an *aggregate optimization problem* or *consensus optimization* problem for the collection of agents $1, \ldots, K$.

Example: Least-Squares Fusion

We specialize the discussion again to the least-squares problem. For the local models we would have:

$$w_k^\star = \arg\min_{w\in\mathbb{R}^M} \left\{ \frac{1}{2N}\sum_{n=1}^{N}\left(\gamma_{k,n} - h_{k,n}^{\mathsf{T}} w\right)^2 + \frac{\rho}{2}\|w\|^2 \right\} \tag{44}$$

where we defined $\rho = \frac{1}{N}\frac{\sigma_v^2}{\sigma_w^2}$ for notational convenience. Since (44) is quadratic over w, we can determine a closed-form expression for $w_k^\star$ by differentiating:

$$-\frac{1}{N}\sum_{n=1}^{N}\gamma_{k,n}h_{k,n} + \left(\frac{1}{N}\sum_{n=1}^{N}h_{k,n}h_{k,n}^{\mathsf{T}}\right) w_k^\star + \rho w_k^\star = 0 \tag{45}$$

Example: Least-Squares Fusion

After grouping terms:

$$
\begin{aligned}
w_k^\star &= \left(\frac{1}{N}\sum_{n=1}^{N} h_{k,n}h_{k,n}^{\mathsf{T}} + \rho I\right)^{-1}\left(\frac{1}{N}\sum_{n=1}^{N}\gamma_{k,n}h_{k,n}\right) \\
&= (H_k + \rho I)^{-1} d_k
\end{aligned} \tag{46}
$$

We defined for compactness:

$$
H_k \triangleq \frac{1}{N}\sum_{n=1}^{N} h_{k,n}h_{k,n}^{\mathsf{T}} \tag{47}
$$

$$
d_k \triangleq \frac{1}{N}\sum_{n=1}^{N}\gamma_{k,n}h_{k,n} \tag{48}
$$

Example: Least-Squares Fusion

For the aggregate problem (43) we can find analogously:

$$\begin{aligned} w^\star &= \left(\frac{1}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N} h_{k,n}h_{k,n}^{\mathsf{T}} + \frac{\rho}{K}I\right)^{-1}\left(\frac{1}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N}\gamma_{k,n}h_{k,n}\right) \\ &= \left(H + \frac{\rho}{K}I\right)^{-1} d \end{aligned} \tag{49}$$

where we defined:

$$H = \frac{1}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N} h_{k,n}h_{k,n}^{\mathsf{T}} = \frac{1}{K}\sum_{k=1}^{K} H_k \tag{50}$$

$$d = \frac{1}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N}\gamma_{k,n}h_{k,n} = \frac{1}{K}\sum_{k=1}^{K} d_k \tag{51}$$

Exchanging Processed Quantities

Returning to the quadratic problem, note that:

$$\Big(H + \frac{\rho}{K} I\Big) w^\star = d = \frac{1}{K} \sum_{k=1}^{K} d_k \overset{(46)}{=} \frac{1}{K} \sum_{k=1}^{K} (H_k + \rho I) w_k^\star \tag{52}$$

It then follows that $w^\star$ can be evaluated equivalently as:

$$w^\star = \left(\frac{1}{K} \sum_{k=1}^{K} H_k + \frac{\rho}{K} I \right)^{-1} \left(\frac{1}{K} \sum_{k=1}^{K} (H_k + \rho I) w_k^\star \right) \tag{53}$$

This means that rather than communicate raw data $h_{k,n}$ at the fusion center, and subsequently compute $w^\star$ via (49), we can instead locally compute H_k and $w_k^\star$, and transmit only these processed quantities to the fusion center, and then apply (53).

- The first approach requires the communication of $K(M+1)N$ scalar quantities.
- The second approach requires K times the exchange of a matrix of size M^2 and a vector of size M, hence a cost of $K(M+1)M$.

Recap and Outlook

- We used Bayesian techniques to motivate empirical and expected risk minimization problems of the form:

$$w^\star = \arg\min_w \sum_{n=1}^{N} Q(w; x_n) + R(w) \triangleq \arg\min_w J(w) \tag{54}$$

$$w^o = \arg\min_w \mathbb{E}_{\boldsymbol{x}} Q(w; \boldsymbol{x}) \triangleq \arg\min_w J(w) \tag{55}$$

- We also saw how same line of reasoning motivates aggregate or consensus optimization problems of the form:

$$J(w) = \frac{1}{K} \sum_{k=1}^{K} J_k(w) \tag{56}$$

- In the case of linear models with Gaussian noise, this led to quadratic problems where we could do all calculations in closed-form.
- For more general objective functions $J(w)$, we will have to appeal to iterative methods.

Deterministic Optimization

We are interested in optimizing a general objective:

$$w^\star \triangleq \arg\min_{w} J(w) \tag{57}$$

- We need to assume some regularity conditions. For the purposes of this course, we will focus of ν-strongly convex objectives with δ-Lipschitz gradients.
- It is possible to relax both the convexity and smoothness conditions on the risks, but many of the themes and conclusions will continue to hold. So to avoid unnecessary complications, and focus on key ideas, we will work with convex and smooth objectives.

ν-Strong Convexity

We will say that the risk is ν-strongly convex if:

$$J(w_2) \geq J(w_1) + \nabla J(w)^{\mathsf{T}}(w_2 - w_1) + \frac{\nu}{2}\|w_2 - w_1\|^2 \tag{58}$$

for some positive parameter $\nu \geq 0$. Note that strong convexity is a stronger condition than convexity, and in particular we recover convexity when $\nu = 0$. Now suppose we have found, by whatever means, a point satisfying the first-order condition $\nabla J(w^\star) = 0$. Then, it follows from (58) that for any $w \in \operatorname{dom} J$:

$$\begin{aligned} J(w) &\geq J(w^\star) + \nabla J(w^\star)^{\mathsf{T}}(w - w^o) + \frac{\nu}{2}\|w - w^o\|^2 \\ &= J(w^o) + \frac{\nu}{2}\|w - w^o\|^2 \end{aligned} \tag{59}$$

Hence, w^o is the unique minimizing argument of $J(w)$.

δ-Lipschitz Gradients

We will say that $J(w)$ is smooth if it satisfies:

$$J(w_2) \leq J(w_1) + \nabla J(w_1)^{\mathsf{T}}(w_2 - w_1) + \frac{\delta}{2}\|w_2 - w_1\|^2 \tag{60}$$

Condition (60) is analogous to (58), except that now we impose an *upper* bound on the growth above the tangent hyperplane, where δ controls the rate of growth. This implies the statement:

$$\|\nabla J(w_2) - \nabla J(w_1)\| \leq \delta\|w_2 - w_1\| \tag{61}$$

In other words, the smoothness condition (60) translates into a Lipschitz condition on the gradients of $J(w)$.

Gradient Descent

We can then pursue

$$w^\star \triangleq \arg\min_w J(w) \tag{62}$$

by iterating from some initial point $w_0 \in \mathbb{R}^M$:

$$w_i = w_{i-1} - \mu \nabla J(w_{i-1}) \tag{63}$$

which is known as the gradient descent algorithm. The name comes from the fact that:

$$\begin{aligned} J(w_i) &\overset{(60)}{\leq} J(w_{i-1}) + \nabla J(w_{i-1})^{\mathsf{T}}(w_i - w_{i-1}) + \frac{\delta}{2}\|w_i - w_{i-1}\|^2 \\ &\overset{(63)}{=} J(w_{i-1}) + \nabla J(w_{i-1})^{\mathsf{T}}(-\mu \nabla J(w_{i-1})) + \frac{\delta}{2}\|\mu \nabla J(w_{i-1})\|^2 \\ &= J(w_{i-1}) - \mu\left(1 - \mu\frac{\delta}{2}\right)\|\nabla J(w_{i-1})\|^2 \\ &\overset{(a)}{\leq} J(w_{i-1}) \end{aligned} \tag{64}$$

Visualization of Gradient Descent

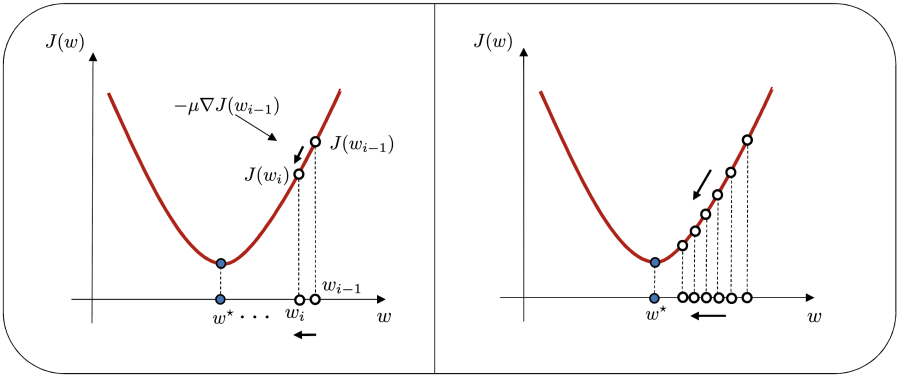


Figure: The panel on the left shows the mechanics of one update step where w_{i-1} is updated in the direction of the minimizer $w^\star$. The panel on the right shows the result of several successive steps with the iterates approaching $w^\star$.

Example: Logistic Regression

Quadratic loss functions are not appropriate when the label γ in an inference problem takes discrete values $\gamma = \pm 1$. In this case, we a more appropriate choice is the regularized logistic risk function defined by

$$J(w) = \frac{\rho}{2}\|w\|^2 \; + \; \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{-\gamma_n h_n^{\mathsf{T}} w}\right) \tag{65}$$

for which

$$\nabla J(w) = \rho w \; - \; \frac{1}{N}\sum_{n=1}^{N} \frac{\gamma_n h_n}{1 + e^{\gamma_n h_n^{\mathsf{T}} w}} \tag{66}$$

Example: Logistic Regression

The gradient-descent recursion (63) in this case leads to

$$w_i = (1 - \mu\rho)\, w_{i-1} + \mu \left(\frac{1}{N} \sum_{n=1}^{N} \frac{\gamma_n h_n}{1 + e^{\gamma_n h_n^{\mathsf{T}} w_{i-1}}} \right), \quad i \geq 0 \tag{67}$$

One can interpret the logistic risk function as arising from a Bayes' optimal estimate for $\boldsymbol{w}$ under the statistical model:

$$f_{\boldsymbol{\gamma}|\boldsymbol{h},\boldsymbol{w}}(\gamma|h, w) = \frac{1}{1 + e^{-\gamma h^{\mathsf{T}} w}} \tag{68}$$

with a Gaussian prior on $\boldsymbol{w}$, though the details of this derivation are not relevant to this course. Once we have found $w^\star$, we can then perform inference on unlabelled data $\boldsymbol{h}$ by computing $\operatorname{sign}\left(\boldsymbol{h}^{\mathsf{T}} w^\star\right)$ as illustrated on the next slide. Features lying on one side of the hyperplane whose normal direction is $w^\star$ belong to class $+1$ while the other features belong to class -1.

Example: Logistic Regression

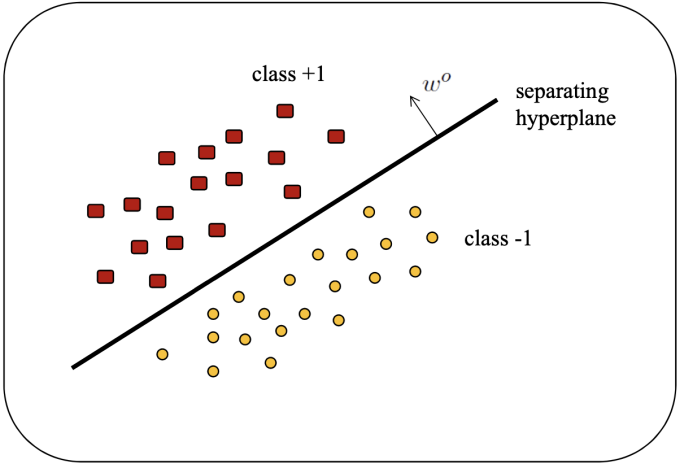


Figure: A logistic regression problem, with separating hyperplane defined by the parameter $w^\star$.

Example: Logistic Regression

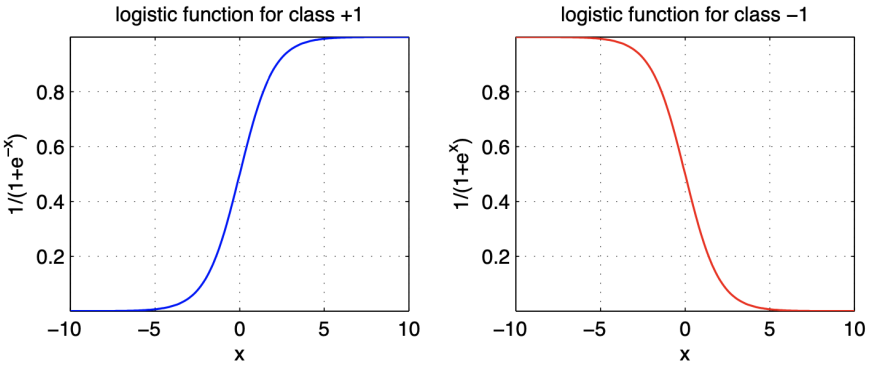


Figure: The left figure shows the logistic function (68) when $\gamma = +1$ as a function of $x = h^{\mathsf{T}}w$, the right figure shows the same for $\gamma = -1$. This model imposes that it is likely that $\boldsymbol{\gamma}$ and $\boldsymbol{h}^{\mathsf{T}}\boldsymbol{w}$ have the same sign, with the likelihood increasing with the absolute value of $\boldsymbol{h}^{\mathsf{T}}\boldsymbol{w}$.

Convergence Analysis

Convergence Of Gradient-Descent

Consider the gradient-descent recursion (63) where $J(w)$ is a ν-strongly convex function with δ-Lipschitz gradients. Introduce the error vector $\widetilde{w}_i = w^\star - w_i$, which measures the difference between the $i-$th iterate and the global minimizer of $J(w)$. If the step-size μ satisfies:

$$0 < \mu < 2\nu/\delta^2 \tag{69}$$

then w_i converges exponentially fast to $w^\star$ in the sense that

$$\|\widetilde{w}_i\|^2 \;\le\; \lambda\,\|\widetilde{w}_{i-1}\|^2, \;\; i \ge 0 \tag{70}$$

where $\lambda \;=\; 1 - 2\mu\nu + \mu^2\delta^2 \;\in [0,1)$. It also holds that the risk value converges exponentially fast as follows

$$J(w_i) - J(w^\star) \;\le\; \delta\lambda^i \|\widetilde{w}_0\|^2 \;=\; O(\lambda^i) \tag{71}$$

Rate of Convergence

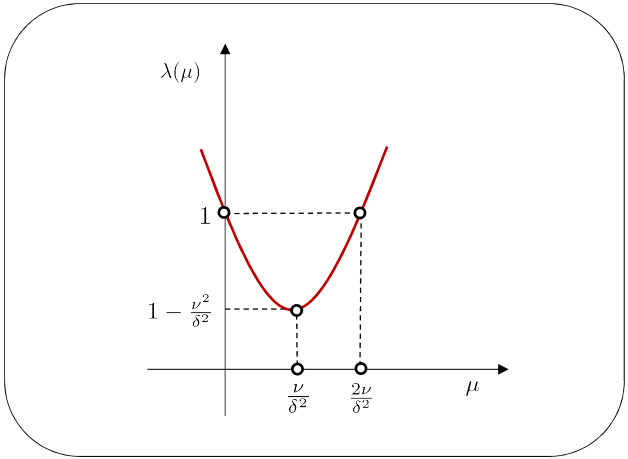


Figure: Plot of the function $\lambda(\mu) = 1 - 2\nu\mu + \mu^2\delta^2$. It shows that the function $\lambda(\mu)$ assumes values below one in the range $0 < \mu < 2\nu/\delta^2$.

Transient and steady-state error

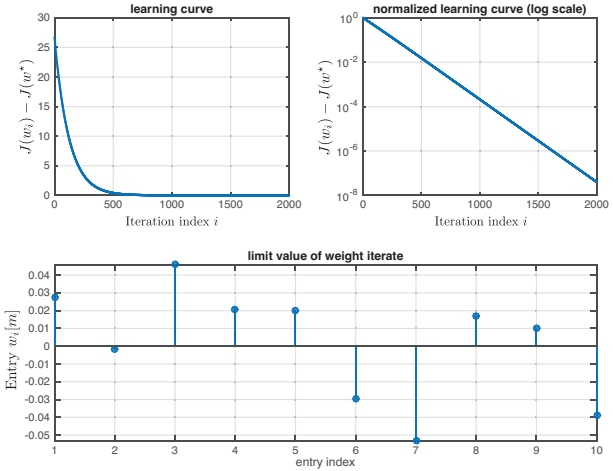


Figure: Learning curves $J(w_i)$ relative to the minimum risk value $J(w^\star)$ in linear scale (on the left) and in normalized logarithmic scale (on the right). (*Bottom*) Limiting value of w_i, which tends to $w^\star$.

Recap and Outlook

- We have seen how the gradient algorithm can iteratively minimize general objective functions.
- However, the recursions relied on exact gradients. In practice, this can be unavailable or costly, particularly in many applications of multi-agent systems.
- We will now see how we can tackle these issues with stochastic gradient algorithms.

Motivation for Stochastic Gradients

We will continue to study optimization problems of the form:

$$\min_{w \in \mathbb{R}^M} J(w) \tag{72}$$

In the previous section, we examined the performance of the gradient-descent algorithm, which takes the form:

$$w_i = w_{i-1} - \mu \nabla J(w_{i-1}) \tag{73}$$

In practice, many times computing $\nabla J(w_{i-1})$ is costly or infeasible.

Example: Expected Risk Minimization

Suppose the risk $J(w)$ is defined as the expected value of some loss function $Q(w;\boldsymbol{x})$ over the distribution of the data $\boldsymbol{x}$:

$$J(w) \triangleq \mathbb{E}_{\boldsymbol{x}} Q(w;\boldsymbol{x}) \tag{74}$$

It is clear that the evaluation of $J(w)$ requires knowledge of the distribution of $\boldsymbol{x}$, as well as the evaluation of its gradient vector since:

$$\nabla J(w) = \nabla \left(\mathbb{E}_{\boldsymbol{x}} Q(w;\boldsymbol{x})\right) \tag{75}$$

Suppose at time i, we observe a single sample $\boldsymbol{x}_i$ from the distribution of $\boldsymbol{x}$. Then, we can construct the following instantaneous approximation for the gradient vector:

$$\widehat{\nabla J}(w;\boldsymbol{x}_i) \triangleq \nabla Q(w;\boldsymbol{x}_i) \tag{76}$$

and use it in (73):

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu \widehat{\nabla J}(\boldsymbol{w}_{i-1};\boldsymbol{x}_i) = \boldsymbol{w}_{i-1} - \mu \nabla Q(\boldsymbol{w}_{i-1};\boldsymbol{x}_i) \tag{77}$$

Observe that the stochastic gradient approximation $\nabla Q(\boldsymbol{w}_{i-1};\boldsymbol{x}_i)$ is a function of the random variable $\boldsymbol{x}_i$, and is hence random itself.

Example: Empirical Risk Minimization

A second class of problems that appear in learning contexts are empirical risk minimization problems of the form:

$$J(w) \triangleq \frac{1}{N} \sum_{n=1}^{N} Q(w; x_n) \tag{78}$$

where the set of points $\{x_n\}_{n=1}^{N}$ corresponds to a batch of N independent realizations of the random variable $\boldsymbol{x}$. In contrast to the previous example, where evaluation of the gradient $\nabla J(w)$ is infeasible, for empirical risk minimization problems, it is in principle possible to iterate:

$$w_i = w_{i-1} - \mu \nabla J(w_{i-1}) = w_{i-1} - \frac{\mu}{N} \sum_{n=1}^{N} \nabla Q(w_{i-1}; x_n) \tag{79}$$

The drawback now is that every iteration requires the evaluation of N gradients of the risk $\nabla Q(w_{i-1}; x_n)$. This results in a costly procedure, especially in large-scale settings, where the sample size N is large.

Example: Empirical Risk Minimization

We may motivate a stochastic approximation for the true gradient appearing in (79) as follows. At any time i, we sample from the set of data points $\{x_n\}_{n=1}^N$ uniformly at random. We may model this procedure by defining a random index $\boldsymbol{n}_i$, with uniform distribution:

$$\boldsymbol{n}_i = \begin{cases} 1, & \text{with probability } \frac{1}{N}, \\ 2, & \text{with probability } \frac{1}{N}, \\ \vdots & \\ N, & \text{with probability } \frac{1}{N}. \end{cases} \tag{80}$$

Then $x_{\boldsymbol{n}_i}$ denotes the sample picked randomly from $\{x_n\}_{n=1}^N$ at time i. We use this sample to construct a stochastic approximation for the gradient as

$$\widehat{\nabla J}(w; x_{\boldsymbol{n}_i}) \triangleq \nabla Q(w; x_{\boldsymbol{n}_i}) \tag{81}$$

This construction results in a stochastic variant of recursion (79):

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu \nabla Q(\boldsymbol{w}_{i-1}, x_{\boldsymbol{n}_i}) \tag{82}$$

Example: Empirical Risk Minimization

Note further that if we let $\boldsymbol{x}_i^{\text{emp}} \triangleq x_{\boldsymbol{n}_i}$, it follows that:

$$J(w) \triangleq \frac{1}{N}\sum_{n=1}^{N} Q(w;x_n) = \mathbb{E}_{\boldsymbol{x}_i^{\text{emp}}} Q(w;\boldsymbol{x}_i^{\text{emp}}) = \mathbb{E}_{\boldsymbol{x}^{\text{emp}}} Q(w;\boldsymbol{x}^{\text{emp}}) \tag{83}$$

In other words, the empirical risk minimization problem (78) is a special case of the expected risk minimization problem (74), where we define a new random variable $\boldsymbol{x}^{\text{emp}} \triangleq x_{\boldsymbol{n}}$, which follows a uniform empirical distribution over the sample batch of data $\{x_n\}_{n=1}^N$. Note that for any finite N, there will be a difference between $\boldsymbol{x}$, the underlying random variable that led to the batch of samples $\{x_n\}_{n=1}^N$, and $\boldsymbol{x}^{\text{emp}}$, which is sampled from $\{x_n\}_{n=1}^N$.

Stochastic Gradient Approximations

Given a risk function $J(w)$ we denote by $\widehat{\nabla J}(w)$ a stochastic gradient approximation for the true gradient $\nabla J(w)$ if $\widehat{\nabla J}(w)$ can be evaluated solely from data available to the algorithm and

$$\widehat{\nabla J}(w) \approx \nabla J(w) \tag{84}$$

in some sense. On occasion, we may wish to make precise the exact data used to compute $\widehat{\nabla J}(w)$, in which case we may for example write:

$$\widehat{\nabla J}(w; \boldsymbol{x}_i) \approx \nabla J(w) \tag{85}$$

to emphasize that $\widehat{\nabla J}(w; \boldsymbol{x}_i)$ is constructed based on a realization of the random variable $\boldsymbol{x}_i$. As a general rule, we will employ $\widehat{\nabla J}(w)$ whenever the data used to construct the gradient approximation is either clear from the context (to lighten the notation), or irrelevant to the discussion (to keep the results general).

Gradient Noise

Motivated by the discussion so far, we will now study generic expected risk minimization problems of the form:

$$\arg\min_{w\in\mathbb{R}^M} J(w) \triangleq \arg\min_{w\in\mathbb{R}^M} \mathbb{E}_{\boldsymbol{x}} Q(w;\boldsymbol{x}) \tag{86}$$

with the understanding that this formulation covers empirical risk minimization problems as laid out in the previous example. We will consider a stochastic gradient algorithm with a generic gradient approximation:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu\widehat{\nabla J}(\boldsymbol{w}_{i-1}) \tag{87}$$

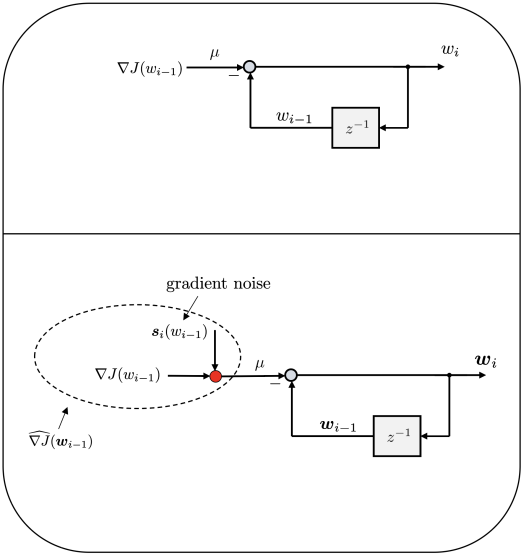
It is instructive to reformulate the recursion as (see Fig. 59):

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \tag{88}$$

in terms of the *gradient noise* term defined by

$$\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) = \widehat{\nabla J}(\boldsymbol{w}_{i-1}) - \nabla J(\boldsymbol{w}_{i-1}) \tag{89}$$

Stochastic Gradient Algorithms as Perturbed Systems



Gradient Noise Modeling Conditions

The gradient approximations $\widehat{\nabla J}(\boldsymbol{w}_{i-1})$ are unbiased conditioned on the iterate $\boldsymbol{w}_{i-1}$. Specifically:

$$\mathbb{E}\left\{\widehat{\nabla J}(\boldsymbol{w}_{i-1})|\,\boldsymbol{w}_{i-1}\right\} = \nabla J(\boldsymbol{w}_{i-1}) \Longleftrightarrow \mathbb{E}\left\{\boldsymbol{s}_i(\boldsymbol{w}_{i-1})|\,\boldsymbol{w}_{i-1}\right\} = 0 \tag{90}$$

Unbiased gradient estimates ensure that the gradient approximation $\widehat{\nabla J}(\boldsymbol{w}_{i-1})$ is, on average, a good estimate for the descent direction. Unbiased estimates are good estimates, but only if their variance is bounded. We hence impose an additional condition:

$$\begin{aligned}\mathbb{E}\left\{\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2|\,\boldsymbol{w}_{i-1}\right\} \leq\ & \alpha^2\|\nabla J(\boldsymbol{w}_{i-1})\|^2 + \beta^2\|w^o - \boldsymbol{w}_{i-1}\|^2 \\ & + \gamma^2\left(J(\boldsymbol{w}_{i-1}) - J^o\right) + \sigma^2\end{aligned} \tag{91}$$

This second condition imposes a *relative* bound on the variance of the gradient approximation because we allow the right-hand side of (91) to grow with various measures of suboptimality.

On the generality of condition (91)

We note that the variance condition (91) is deliberately chosen to be general by allowing for relative components proportional to $\|\nabla J(\boldsymbol{w}_{i-1})\|^2$, $\|w^o - \boldsymbol{w}_{i-1}\|^2$, and $J(\boldsymbol{w}_{i-1}) - J^o$ through the constants α, β and γ, respectively. It is useful to observe that for objectives with δ-Lipschitz gradients, it holds that:

$$\frac{1}{2\delta}\|\nabla J(\boldsymbol{w}_{i-1})\|^2 \leq J(\boldsymbol{w}_{i-1}) - J^o \leq \frac{\delta}{2}\|w^o - \boldsymbol{w}_{i-1}\|^2 \tag{92}$$

As such, any gradient approximation that satisfies (91) will also satisfy:

$$\mathbb{E}\left\{\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 | \, \boldsymbol{w}_{i-1}\right\} \leq \beta^2\|w^o - \boldsymbol{w}_{i-1}\|^2 + \sigma^2 \tag{93}$$

for some β^2 that is suitably adjusted. For this reason, at several points in our presentation, and in order to lighten the notation, we will set $\alpha = \gamma = 0$ and work with the simplified condition (93). Nevertheless, allowing for more granular characterization of the relative components in (91) is useful depending on the setting, so we allow for it when it doesn't result in unnecessary complications.

Elementary Gradient Approximations

Perhaps the most commonly employed gradient approximation is the one obtained from the expression $\nabla J(w) \triangleq \nabla \{\mathbb{E}_{\boldsymbol{x}} Q(w; \boldsymbol{x})\}$ by dropping the expectation and using the loss value at the sample $\boldsymbol{x}_i$ available at time i. This results in:

$$\widehat{\nabla J}^{\,\mathrm{ele}}(w) \triangleq \nabla Q(w; \boldsymbol{x}_i) \tag{94}$$

We will be referring to such approximations as *elementary* stochastic gradient approximations. They are the most common and simplest constructions, and we will see that many more elaborate constructions can be "built" from elementary gradient approximations. When we wish to emphasize the fact that the construction is of the elementary type, or to distinguish it from other constructions, we will attach the superscript "ele" to the relevant quantities. For example, $\widehat{\nabla J}^{\,\mathrm{ele}}(w)$ refers to the elementary gradient approximation constructed according to (94), while $\boldsymbol{s}_i^{\mathrm{ele}}(w)$ refers to the resulting gradient noise with parameters $\alpha^2_{\mathrm{ele}}, \beta^2_{\mathrm{ele}}, \gamma^2_{\mathrm{ele}}, \sigma^2_{\mathrm{ele}}$. When the context is clear, or we wish to preserve generality of the statements, we will omit "ele".

Example: Mean-square error learning

Let us consider the linear regression model:

$$\boldsymbol{\gamma} = \boldsymbol{h}^{\mathsf{T}} w^o + \boldsymbol{v} \tag{95}$$

where $\boldsymbol{h} \in \mathbb{R}^M$ denotes the regressor and $\boldsymbol{\gamma}$ denotes the observation. The measurement noise is denoted by $\boldsymbol{v}$ and assumed to be zero-mean and independent of $\boldsymbol{h}$. Given observations $\boldsymbol{x} \triangleq \mathrm{col}\{\boldsymbol{h}, \boldsymbol{\gamma}\}$, we may find the unknown model w^o by solving:

$$w^o = \underset{w \in \mathbb{R}^M}{\arg\min} \frac{1}{2} \mathbb{E} \|\boldsymbol{\gamma} - \boldsymbol{h}^{\mathsf{T}} w\|^2 \tag{96}$$

Let $Q(w; \boldsymbol{x}) = Q(w; \boldsymbol{h}, \boldsymbol{\gamma}) = \frac{1}{2}\|\boldsymbol{\gamma} - \boldsymbol{h}^{\mathsf{T}} w\|^2$. By differentiating, we find:

$$\nabla J(w) = \mathbb{E} \left\{ -\boldsymbol{h} \left(\boldsymbol{\gamma} - \boldsymbol{h}^{\mathsf{T}} w \right) \right\} = -r_{\boldsymbol{h}\boldsymbol{\gamma}} + R_{\boldsymbol{h}} w \tag{97}$$

where we defined $r_{\boldsymbol{h}\boldsymbol{\gamma}} = \mathbb{E}\boldsymbol{h}\boldsymbol{\gamma}$ and $R_{\boldsymbol{h}} \triangleq \mathbb{E}\boldsymbol{h}\boldsymbol{h}^{\mathsf{T}}$.

Example: Mean-square error learning

We can then construct:

$$\widehat{\nabla J}(w) = \nabla Q(w; \boldsymbol{x}_i) = -\boldsymbol{h}_i \left(\boldsymbol{\gamma}_i - \boldsymbol{h}_i^{\mathsf{T}} w\right) = -\boldsymbol{h}_i \boldsymbol{\gamma}_i + \boldsymbol{h}_i \boldsymbol{h}_i^{\mathsf{T}} w \tag{98}$$

so that the gradient noise $\boldsymbol{s}_i(w)$ is given by

$$\boldsymbol{s}_i(w) = -\boldsymbol{h}_i \boldsymbol{\gamma}_i + \boldsymbol{h}_i \boldsymbol{h}_i^{\mathsf{T}} w + r_{\boldsymbol{h}\boldsymbol{\gamma}} - R_{\boldsymbol{h}} w \tag{99}$$

It can be verified that $\boldsymbol{s}_i(\boldsymbol{w}_{i-1})$ is zero-mean and satisfies the variance bound with $\alpha^2 = \gamma^2 = 0$ and:

$$\beta^2 \triangleq \mathbb{E} \left\| \boldsymbol{h}_i \boldsymbol{h}_i^{\mathsf{T}} - R_{\boldsymbol{h}} \right\|^2 \tag{100}$$

$$\sigma^2 \triangleq \sigma_{\boldsymbol{v}}^2 \mathrm{Tr}\left(R_{\boldsymbol{h}}\right) \tag{101}$$

Example: Logistic Regression

In this case, the labels $\boldsymbol{\gamma}$ are sampled from a discrete distribution, say $\boldsymbol{\gamma} = \pm 1$, while

$$Q(w; \boldsymbol{h}_i, \boldsymbol{\gamma}_i) \triangleq \ln\left(1 + e^{-\boldsymbol{\gamma}_i \boldsymbol{h}_i^{\mathsf{T}} w}\right) + \frac{\rho}{2}\|w\|^2 \tag{102}$$

$$J(w) \triangleq \mathbb{E}\, Q(w; \boldsymbol{h}_i, \boldsymbol{\gamma}_i) = \mathbb{E} \ln\left(1 + e^{-\boldsymbol{\gamma}_i \boldsymbol{h}_i^{\mathsf{T}} w}\right) + \frac{\rho}{2}\|w\|^2 \tag{103}$$

Then, it follows that:

$$\begin{aligned} \nabla Q(w; \boldsymbol{h}_i, \boldsymbol{\gamma}_i) = & -\frac{\boldsymbol{\gamma}_i \boldsymbol{h}_i}{1 + e^{\boldsymbol{\gamma}_i \boldsymbol{h}_i^{\mathsf{T}} w}} + \rho w \\ \nabla J(w) = & -\mathbb{E}\left\{\frac{\boldsymbol{\gamma}_i \boldsymbol{h}_i}{1 + e^{\boldsymbol{\gamma}_i \boldsymbol{h}_i^{\mathsf{T}} w}}\right\} + \rho w \end{aligned} \tag{104}$$

Hence, we can construct $\widehat{\nabla J}(w) = \nabla Q(w; \boldsymbol{h}_i, \boldsymbol{\gamma}_i)$ and immediately verify the zero-mean condition (90).

Example: Logistic Regression

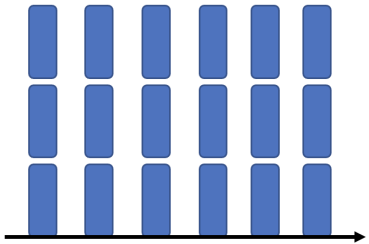
For the variance bound, we can verify that:

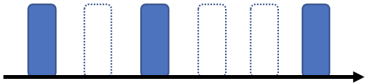
$$\mathbb{E}\left\{\|\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2|\,\boldsymbol{w}_{i-1}\right\} \leq \mathbb{E}\|\boldsymbol{h}_i\|^2 = \mathrm{Tr}\,(R_{\boldsymbol{h}}) \tag{105}$$

where we defined $R_{\boldsymbol{h}} = \mathbb{E}\boldsymbol{h}\boldsymbol{h}^{\mathsf{T}}$. Hence, we know that the elementary logistic regression recursion satisfies the gradient noise conditions with $\alpha^2 = \beta^2 = \gamma^2 = 0$ and $\sigma^2 = \mathrm{Tr}\,(R_{\boldsymbol{h}})$.

Variants of Stochastic Gradient Descent

- We might receive multiple samples at a time, or sometimes receive no samples at all.
- The stochastic approximation framework is flexible to cover these variants.





$$\boldsymbol{w}_{k,i} = \boldsymbol{w}_{k,i-1} - \frac{\mu}{3}\sum_{b=1}^{3}\nabla Q(\boldsymbol{w}_{k,i-1};\boldsymbol{x}_{k,i,b})$$

$$\boldsymbol{w}_{k,i} = \begin{cases} \boldsymbol{w}_{k,i-1} - \frac{\mu}{\pi}\nabla Q(\boldsymbol{w}_{k,i-1};\boldsymbol{x}_{k,i}) & \text{w.p. } \pi, \\ \boldsymbol{w}_{k,i-1} & \text{w.p. } 1-\pi. \end{cases}$$

Mini-Batch SGD

Suppose that instead of being provided with a single data point $\boldsymbol{x}_i$ at any given time i, we have access to B independent samples $\{\boldsymbol{x}_{b,i}\}_{b=1}^{B}$. We can then compute:

$$\widehat{\nabla J}\left(\boldsymbol{w}_{i-1};\{\boldsymbol{x}_{b,i}\}_{b=1}^{B}\right) \triangleq \frac{1}{B}\sum_{b=1}^{B}\nabla Q(\boldsymbol{w}_{i-1};\boldsymbol{x}_{b,i}) \tag{106}$$

It is straightforward to verify that:

$$\mathbb{E}\left\{\widehat{\nabla J}(\boldsymbol{w}_{i-1};\{\boldsymbol{x}_{b,i}\}_{b=1}^{B})|\,\boldsymbol{w}_{i-1}\right\} = \nabla J(\boldsymbol{w}_{i-1}) \tag{107}$$

Using the independence of samples, we have for the resulting gradient noise process:

$$\mathbb{E}\left\{\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2|\,\boldsymbol{w}_{i-1}\right\} \leq \alpha_B^2\|\nabla J(\boldsymbol{w}_{i-1})\|^2 + \beta_B^2\|w^o - \boldsymbol{w}_{i-1}\|^2 + \gamma_B^2\left(J(\boldsymbol{w}_{i-1}) - J^o\right) + \sigma_B^2 \tag{108}$$

with

$$\alpha_B^2 \triangleq \frac{\alpha_{\text{ele}}^2}{B}, \qquad \beta_B^2 \triangleq \frac{\beta_{\text{ele}}^2}{B}, \qquad \gamma_B^2 \triangleq \frac{\gamma_{\text{ele}}^2}{B}, \qquad \sigma_B^2 \triangleq \frac{\sigma_{\text{ele}}^2}{B} \tag{109}$$

Stochastic Gradient with Missing Samples

We define an i.i.d. Bernoulli random indicator variable:

$$\mathbb{I}_i = \begin{cases} 1, & \text{with probability } \pi, \\ 0, & \text{otherwise.} \end{cases} \tag{110}$$

where $\mathbb{I}_i$ indicates whether we receive a sample $\boldsymbol{x}_i$ at time i or not. We can then construct:

$$\widehat{\nabla J}(w) \triangleq \begin{cases} \dfrac{1}{\pi}\nabla Q(w;\boldsymbol{x}_i), & \text{if } \mathbb{I}_i = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{111}$$

Note that in the case when data is available, the gradient approximation $\frac{1}{\pi}\nabla Q(w;\boldsymbol{x}_i)$ is scaled by $\frac{1}{\pi}$. And since $0 < \pi \leq 1$, this results in larger steps being taken when data is available to compensate for the fact that with probability $1-\pi$, no update occurs. This scaling ensures that we continue to have an unbiased approximation:

$$\mathbb{E}\widehat{\nabla J}(w) = \mathbb{E}\left\{\nabla Q(w;\boldsymbol{x}_i)\right\} = \nabla J(w) \tag{112}$$

Stochastic Gradient with Missing Samples

For the gradient noise variance, we have:

$$\begin{aligned}&\mathbb{E}\left\{\|\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2|\,\boldsymbol{w}_{i-1}\right\}\\&\leq \alpha_{\text{asy}}^2\|\nabla J(\boldsymbol{w}_{i-1})\|^2+\beta_{\text{asy}}^2\|w^o-\boldsymbol{w}_{i-1}\|^2+\gamma_{\text{asy}}^2\left(J(\boldsymbol{w}_{i-1})-J^o\right)+\sigma_{\text{asy}}^2\end{aligned}\tag{113}$$

with:

$$\alpha_{\text{asy}}^2\triangleq\frac{\alpha_{\text{ele}}^2+1-\pi}{\pi};\;\beta_{\text{asy}}^2\triangleq\frac{\beta_{\text{ele}}^2}{\pi};\;\gamma_{\text{asy}}^2\triangleq\frac{\gamma_{\text{ele}}^2}{\pi};\;\sigma_{\text{asy}}^2\triangleq\frac{\sigma_{\text{ele}}^2}{\pi}\tag{114}$$

Convergence Guarantee of Stochastic Gradient Algorithms

Mean-squared deviation of stochastic gradient algorithms

Let the risk function $J(w)$ be ν-strongly convex with δ-Lipschitz gradients. Suppose that we employ a gradient approximation $\widehat{\nabla J}(\cdot)$ satisfying conditions (90) and (91) with constants $\alpha^2, \beta^2, \gamma^2, \sigma^2 \geq 0$. Then, the mean-squared deviation $\mathbb{E}\widetilde{\boldsymbol{w}}_i \triangleq \mathbb{E}\|w^o - \boldsymbol{w}_i\|^2$ of the iterates generated by the stochastic gradient algorithm (87) satisfies:

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \lambda \mathbb{E}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\sigma^2 \tag{115}$$

where we defined

$$\lambda \triangleq 1 - 2\mu\nu + \mu^2\left((1+\alpha^2)\delta^2 + \beta^2 + \gamma^2\frac{\delta}{2}\right) \tag{116}$$

Convergence Guarantee of Stochastic Gradient Algorithms

Mean-squared deviation of stochastic gradient algorithms (2)

Moreover, for sufficiently small step-sizes:

$$\mu \leq \frac{\nu}{(1+\alpha^2)\delta^2 + \beta^2 + \gamma^2 \frac{\delta}{2}} \tag{117}$$

it holds that $\lambda < 1$ and we can iterate (115) to find:

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \lambda^i \|\widetilde{w}_0\|^2 + \frac{\mu\sigma^2}{\nu} \tag{118}$$

For the excess risk, we obtain:

$$\mathbb{E}J(\boldsymbol{w}_i) - J(w^o) \leq \lambda^i \left(\frac{\delta\|\widetilde{w}_0\|^2}{2}\right) + \frac{\mu\delta\sigma^2}{2\nu} \tag{119}$$

Transient and Steady-State Error

We observe that the theorem quantifies both the transient and steady-state behavior of the mean-squared deviation $\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2$ and the excess-risk $\mathbb{E}J(\boldsymbol{w}_i) - J(w^o)$. The steady-state component of an error refers to the part of the error which remains as $i \to \infty$. For the MSD in (118), this corresponds to:

$$\limsup_{i\to\infty} \mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 = \frac{\mu\sigma^2}{\nu} = O(\mu) \tag{120}$$

while the transient component corresponds to $\lambda^i\|\widetilde{w}_0\|^2$. Similarly, for the excess risk, we find in steady-state:

$$\limsup_{i\to\infty} \mathbb{E}J(\boldsymbol{w}_i) - J(w^o) = \frac{\mu\delta\sigma^2}{2\nu} = O(\mu) \tag{121}$$

and identify the transient component as $\lambda^i\left(\frac{\delta\|\widetilde{w}_0\|^2}{2}\right)$.

Alternative Metrics: Iteration Complexity

The iteration complexity of an algorithm is defined as the number of iterations needed for its error to fall below a specified threshold ϵ. We can readily obtain iteration complexities from the characterizations (118) and (119) of the MSD and excess risk respectively. We will illustrate this for the MSD — the argument for the excess risk is analogous. Our objective is to determine the number of iterations i^o needed so that:

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_{i^o}\|^2 \leq \epsilon \tag{122}$$

We can find the IR from (118) by appropriately choosing the step-size μ:

$$i^o \geq \frac{2\sigma^2}{\epsilon}\ln\left(\frac{1}{2\epsilon\|\widetilde{w}_0\|^2}\right) \tag{123}$$

Alternative Metrics: Expected Regret

A second commonly encountered metric for the performance of online learning algorithms is the expected regret, defined the cumulative excess risk encountered up to time T:

$$\text{Regret}_T \triangleq \sum_{i=1}^{T} \mathbb{E}J(\boldsymbol{w}_i) - J(w^o) \tag{124}$$

By suitably choosing the step-size, we can find the expected regret from (119) to be:

$$\sum_{i=1}^{T} (\mathbb{E}J(\boldsymbol{w}_i) - J(w^o)) = O\left(\sqrt{T}\right) \tag{125}$$

Case Study: Mean-Square Error

We illustrate the flexibility of the theorem in combination with the stochastic gradient approximation framework in this chapter by developing a number of variants of the stochastic gradient algorithm along with the corresponding performance bounds. First, let us recall the mean-square error problem, which is given by:

$$J(w) = \frac{1}{2}\mathbb{E}\|\boldsymbol{\gamma} - \boldsymbol{h}^{\mathsf{T}} w\|^2 \tag{126}$$

Given a single pair of observations $\{\boldsymbol{\gamma}_i, \boldsymbol{h}_i\}$, we can construct the elementary gradient approximation:

$$\widehat{\nabla J}^{\text{ele}}(\boldsymbol{w}_{i-1}) = \widehat{\nabla Q}(\boldsymbol{w}_{i-1}; \boldsymbol{h}_i, \boldsymbol{\gamma}_i) = -\boldsymbol{h}_i \left(\boldsymbol{\gamma}_i - \boldsymbol{h}_i^{\mathsf{T}}\, \boldsymbol{w}_{i-1}\right) \tag{127}$$

The resulting stochastic gradient algorithm amounts to:

$$\boldsymbol{w}_i^{\text{ele}} = \boldsymbol{w}_{i-1}^{\text{ele}} - \mu \widehat{\nabla J}^{\text{ele}}(\boldsymbol{w}_{i-1}^{\text{ele}}) = \boldsymbol{w}_{i-1}^{\text{ele}} + \mu \boldsymbol{h}_i \left(\boldsymbol{\gamma}_i - \boldsymbol{h}_i^{\mathsf{T}}\, \boldsymbol{w}_{i-1}^{\text{ele}}\right) \tag{128}$$

Elementary Gradient Approximations

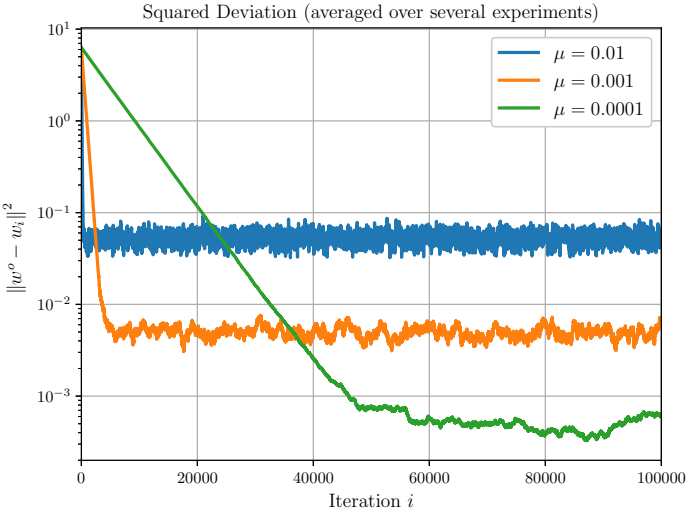
We already computed the resulting gradient noise constants in a previous example to be $\beta^2 = \mathbb{E}\left\|\boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}} - R_{\boldsymbol{h}}\right\|^2$ and $\sigma^2 = \sigma_{\boldsymbol{v}}^2 \mathrm{Tr}\left(R_{\boldsymbol{h}}\right)$ with $\alpha^2 = \gamma^2 = 0$. To find the strong-convexity and Lipschitz parameters ν and δ, we can differentiate (126) twice and find $\nu = \lambda_{\min}\left(R_{\boldsymbol{h}}\right)$ and $\delta = \lambda_{\max}\left(R_{\boldsymbol{h}}\right)$. From our convergence bounds, we can then conclude directly that:

$$\mathbb{E}\left\|w^o - \boldsymbol{w}_i^{\mathrm{ele}}\right\|^2 \leq \lambda_{\mathrm{ele}}^i \|\widetilde{w}_0\|^2 + \frac{\mu\sigma_{\boldsymbol{v}}^2 \mathrm{Tr}\left(R_{\boldsymbol{h}}\right)}{\lambda_{\min}(R_{\boldsymbol{h}})} \tag{129}$$

where:

$$\lambda_{\mathrm{ele}} \triangleq 1 - 2\mu\lambda_{\min}\left(R_{\boldsymbol{h}}\right) + \mu^2\left(\mathbb{E}\left\|\boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}} - R_{\boldsymbol{h}}\right\|^2 + \lambda_{\max}\left(R_{\boldsymbol{h}}\right)\right) \tag{130}$$

Learning Dynamics of Elementary Stochastic Gradient



Mini-Batch Gradient Approximations

If we are instead provide with a mini-batch of observations $\{\boldsymbol{h}_{b,i}, \boldsymbol{\gamma}_{b,i}\}_{b=1}^{B}$ at time i, we can construct:

$$\widehat{\nabla J}^{\mathrm{B}}(\boldsymbol{w}_{i-1}) = \frac{1}{B}\sum_{b=1}^{B} \nabla Q(\boldsymbol{w}_{i-1}; \boldsymbol{h}_{b,i}, \boldsymbol{\gamma}_{b,i}) = -\boldsymbol{h}_{b,i}\left(\boldsymbol{\gamma}_{b,i} - \boldsymbol{h}_{b,i}^{\mathsf{T}}\,\boldsymbol{w}_{i-1}\right) \tag{131}$$

The resulting mini-batch stochastic gradient algorithm amounts to:

$$\boldsymbol{w}_i^{\mathrm{B}} = \boldsymbol{w}_{i-1}^{\mathrm{B}} - \mu\widehat{\nabla J}^{\mathrm{B}}(\boldsymbol{w}_{i-1}^{\mathrm{B}}) = \boldsymbol{w}_{i-1}^{\mathrm{B}} + \frac{\mu}{B}\sum_{b=1}^{B} \boldsymbol{h}_{b,i}\left(\boldsymbol{\gamma}_{b,i} - \boldsymbol{h}_{b,i}^{\mathsf{T}}\,\boldsymbol{w}_{i-1}^{\mathrm{B}}\right) \tag{132}$$

We know from (109) that:

$$\beta_B^2 = \frac{\beta^2}{B} = \frac{\mathbb{E}\left\|\boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}} - R_{\boldsymbol{h}}\right\|^2}{B} \tag{133}$$

$$\sigma_B^2 = \frac{\sigma^2}{B} = \frac{\sigma_{\boldsymbol{v}}^2 \mathrm{Tr}\left(R_{\boldsymbol{h}}\right)}{B} \tag{134}$$

Mini-Batch Gradient Approximations

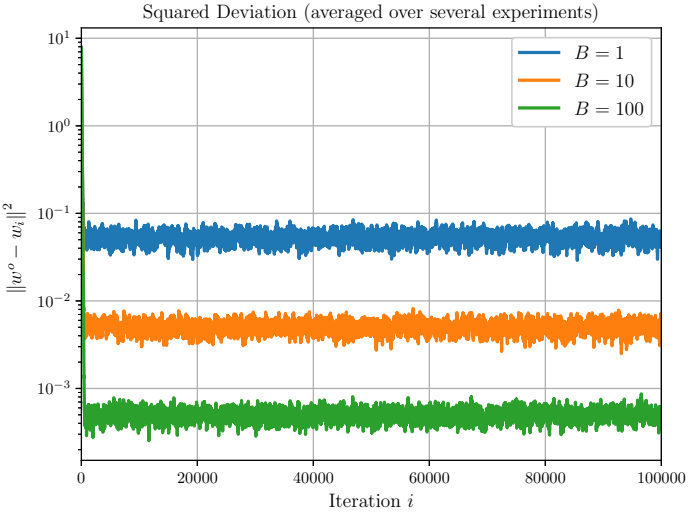
We can then conclude again from (118):

$$\mathbb{E}\left\|w^o - \boldsymbol{w}_i^B\right\|^2 \leq \lambda_B^i \|\widetilde{w}_0\|^2 + \frac{\mu\sigma_{\boldsymbol{v}}^2 \mathrm{Tr}\left(R_{\boldsymbol{h}}\right)}{B\lambda_{\min}(R_{\boldsymbol{h}})} \tag{135}$$

where:

$$\lambda_B \triangleq 1 - 2\mu\lambda_{\min}\left(R_{\boldsymbol{h}}\right) + \mu^2\left(\frac{\mathbb{E}\left\|\boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}} - R_{\boldsymbol{h}}\right\|^2}{B} + \lambda_{\max}\left(R_{\boldsymbol{h}}\right)\right) \tag{136}$$

Learning Dynamics of Mini-Batch Gradient Approximations



Stochastic Gradients with Missing Samples

If we are provided with a pair of observations $\{\boldsymbol{h}_i, \boldsymbol{\gamma}_i\}$ only with probability $0 < \pi \leq 1$, we can construct:

$$\widehat{\nabla J}^{asy}(\boldsymbol{w}_{i-1}) \triangleq \begin{cases} \dfrac{-1}{\pi}\boldsymbol{h}_i\left(\boldsymbol{\gamma}_i - \boldsymbol{h}_i^{\mathsf{T}}\,\boldsymbol{w}_{i-1}\right), & \text{with prob. } \pi, \\ 0, & \text{otherwise.} \end{cases} \tag{137}$$

Then, the resulting stochastic gradient algorithm is given by:

$$\boldsymbol{w}_i^{\text{asy}} = \begin{cases} \boldsymbol{w}_{i-1}^{\text{asy}} + \dfrac{1}{\pi}\boldsymbol{h}_i\left(\boldsymbol{\gamma}_i - \boldsymbol{h}_i^{\mathsf{T}}\,\boldsymbol{w}_{i-1}^{\text{asy}}\right), & \text{with prob. } \pi, \\ \boldsymbol{w}_{i-1}^{\text{asy}}, & \text{otherwise.} \end{cases} \tag{138}$$

with gradient noise constants modified from those of the elementary approximation via (114):

$$\beta_{\text{asy}}^2 = \frac{\beta^2}{\pi} = \frac{\mathbb{E}\left\|\boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}} - R_{\boldsymbol{h}}\right\|^2}{\pi} \tag{139}$$

$$\sigma_{\text{asy}}^2 = \frac{\sigma^2}{\pi} = \frac{\sigma_{\boldsymbol{v}}^2 \text{Tr}\left(R_{\boldsymbol{h}}\right)}{\pi} \tag{140}$$

Stochastic Gradients with Missing Samples

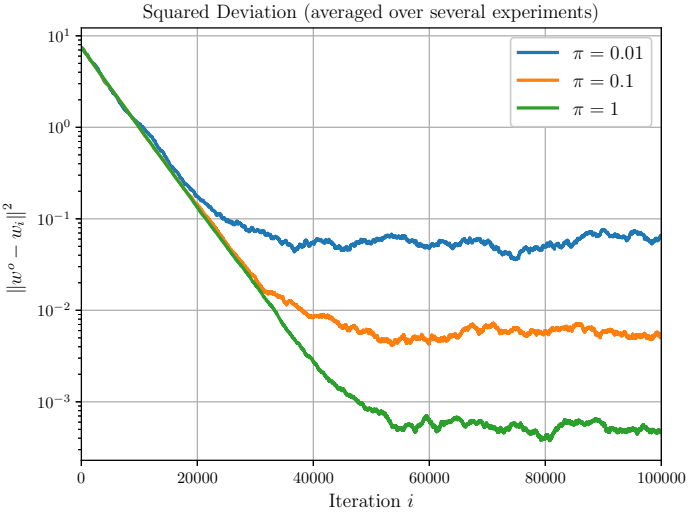
Hence, we find from (118):

$$\mathbb{E}\|w^o - \boldsymbol{w}_i^{\text{asy}}\|^2 \leq \lambda_{\text{asy}}^i \|\widetilde{w}_0\|^2 + \frac{\mu\sigma_{\boldsymbol{v}}^2 \text{Tr}\left(R_{\boldsymbol{h}}\right)}{\pi\lambda_{\min}(R_{\boldsymbol{h}})} \tag{141}$$

where:

$$\lambda_{\text{asy}} \triangleq 1 - 2\mu\lambda_{\min}\left(R_{\boldsymbol{h}}\right) + \mu^2\left(\frac{\mathbb{E}\left\|\boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}} - R_{\boldsymbol{h}}\right\|^2}{\pi} + \lambda_{\max}\left(R_{\boldsymbol{h}}\right)\right) \tag{142}$$

Performance of Stochastic Gradients with Missing Samples



Conclusion

- We saw how:
  - Multi-agent learning problems naturally lead to individual and aggregate risk minimization problems.
  - How stochastic gradient algorithms can solve these risk minimization problems with performance generally limited by a trade-off between quality of gradient approximation, rate of convergence and steady-state error.
- Moving forward, we will develop multi-agent learning algorithms that allow agents to collaboratively learn better/faster/more efficiently than any individual agent.
  - Next lecture we will focus on fusion-center based approaches.
  - From tomorrow, we will look at decentralized architectures.

References and Further Reading

- General references and surveys:
  - A. H. Sayed, *Inference and Learning from Data*, Cambridge University Press, 2022.