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

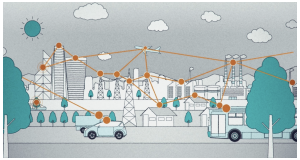


Figure: Sensor networks

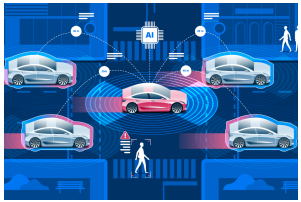


Figure: Autonomous vehicles

Figure: Social network

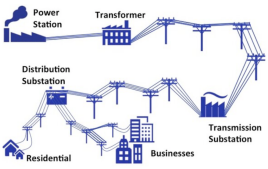


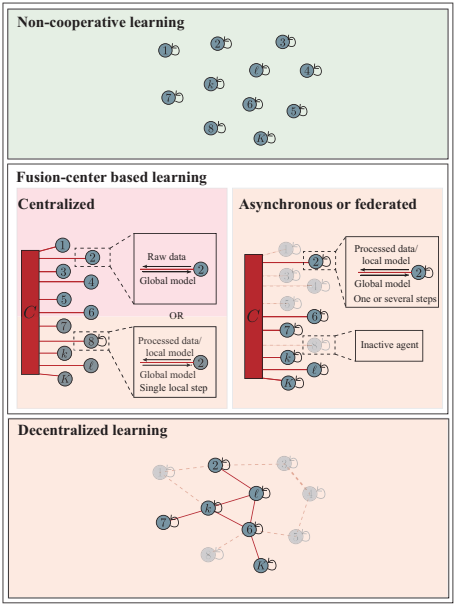
Figure: Power grids

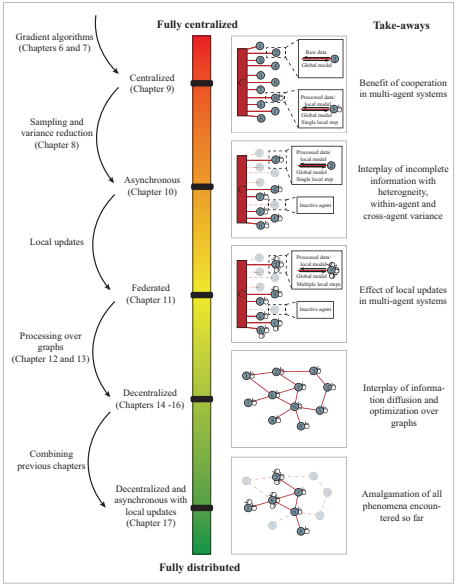
Figure: Mobile devices

Figure: Drone swarms

Structure of this course

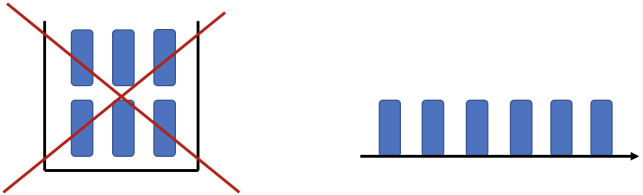
- **Lecture 0:** Background material (pre-reading, available on course website)
  - Matrix theory, random variables, vector calculus.
- **Lecture 1:** Foundations
  - Statistical learning theory, (stochastic) gradient descent, variance reduction.
- **Lecture 2:** Asynchronous and federated Learning
  - Partial participation, local updates, heterogeneity and their impact on performance.
- **Lecture 3:** Graphs and their role decentralized processing
  - Graph spectral theory, averaging and filtering over graphs.
- **Lecture 4:** Algorithms for decentralized optimization and learning
  - Penalty-based (Consensus + innovations and diffusion), primal-dual (EXTRA and Exact diffusion), and gradient-tracking (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.


Non-cooperative learning
Fusion-center based learning
Centralized
Raw data
Global model
OR
Processed data/
local model
Global model
Single local step
Asynchronous or federated
Processed data/
local model
Global model
One or several steps
Inactive agent
Decentralized learning
No exchange
Raw data exchange
Processed data exchange

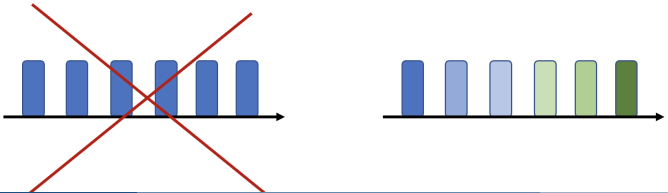

Part I: Background (Chapters 2 through 4)
Fully centralized
Take-aways
Gradient algorithms
(Chapters 6 and 7)
Centralized
(Chapter 9)
Benefit of cooperation
in multi-agent systems
Sampling and
variance reduction
(Chapter 8)
Asynchronous
(Chapter 10)
Interplay of incomplete
information with
heterogeneity,
within-agent and
cross-agent variance
Local updates
Federated
(Chapter 11)
Effect of local updates
in multi-agent systems
Processing over
graphs
(Chapter 12 and 13)
Decentralized
(Chapters 14 -16)
Interplay of informa-
tion diffusion and
optimization over
graphs
Combining
previous chapters
Decentralized and
asynchronous with
local updates
(Chapter 17)
Amalgamation of all
phenomena encoun-
tered so far
Fully distributed
Part V: Advanced Material (Chapters 18 through 24)

Common Themes

- Data is streaming, sequentially available in small quantities.



- Distributions are drifting (slowly).



Aims

Today you will:

- Learn to recognize dynamic environments in various engineering applications.
- Obtain tools for modeling dynamic learning problems.
- Understand how stochastic optimization algorithms, such as stochastic gradient descent can be deployed in dynamic environments.
- Be able to quantify the interplay between algorithm parameters, the level of dynamics, and performance.

Structure of Today's Seminar

- Part I: Static optimization and learning/notation
  - (Convex) optimization and random variables
  - Stochastic gradient approximations, stochastic gradient descent and variants
  - Mean-square contractions
- Part II: Dynamic learning environments
  - Modeling dynamic problems (bounded deviations, random walks)
  - Performance trade-offs in dynamic environments
- Part III: Extensions and applications
  - Federated Learning
  - Decentralized Learning

Great Books on the Topic

- "Introduction to Optimization" by Boris Polyak
- "Convex Optimization" by Stephen Boyd and Lieven Vandenberghe
- "Adaptive Filter Theory" by Simon Haykin
- "Complex Valued Nonlinear Adaptive Filters: Noncircularity, Widely Linear and Neural Models" by Danilo P. Mandic and Vanessa S. L. Goh
- "Adaptive filters" by Ali H. Sayed
- "Adaptation, Learning, and Optimization over Networks" by Ali H. Sayed

Part I: Static Optimization

Optimization

- We consider vector-valued optimization problems:

$$w^o \triangleq \arg\min_{w} J(w) \tag{1}$$

- $w \in \mathbb{R}^M$ is the optimization variable (parameters)
- $w^o \in \mathbb{R}^M$ is the optimal solution
- $J(\cdot) : \mathbb{R}^M \to \mathbb{R}$ is the objective function

Strong Convexity and Lipschitz Gradients

To avoid technical distractions, our analytical results today will be restricted to "well-behaved" functions:

- ν-strongly convex:

$$(\nabla J(x) - \nabla J(y))^{\mathsf{T}} (x - y) \geq \nu \|x - y\|^2 \iff \nabla^2 J(w) \geq \nu I_M \tag{2}$$

- δ-Lipschitz gradients:

$$\|\nabla J(x) - \nabla J(y)\| \leq \delta \|x - y\| \iff \nabla^2 J(w) \leq \delta I_M \tag{3}$$

- Imply quadratic lower and upper bounds on the cost:

$$\frac{\nu}{2} \|w^o - w\|^2 \leq J(w) \leq \frac{\delta}{2} \|w^o - w\|^2 \tag{4}$$

Gradient Descent

- In general, closed-form solutions for $J(w)$ are not available.
- We instead resort to iterative solutions, such as gradient descent:

$$w_i = w_{i-1} - \mu \nabla J(w_{i-1}) \tag{5}$$

- $\nabla J(w_{i-1}) \in \mathbb{R}^M$ denotes the gradient of $J(w_{i-1})$, evaluated at w_{i-1}
- $\mu \in \mathbb{R}^+$ is the step-size (learning rate)

Convergence of Gradient Descent (1)

We show a quick and instrutive proof of convergence. Denote the error:

$$\widetilde{w}_i = w^o - w_i \tag{6}$$

Recall the gradient descent recursion:

$$w_i = w_{i-1} - \mu \nabla J(w_{i-1}) \tag{7}$$

Subtract from w^o to find the error recursion:

$$\widetilde{w}_i = \widetilde{w}_{i-1} + \mu \nabla J(w_{i-1}) - \mu \underbrace{\nabla J(w^o)}_{=0} \tag{8}$$

Convergence of Gradient Descent (2)

Square and expand:

$$\begin{aligned}\|\widetilde{w}_i\|^2 &= \|\widetilde{w}_{i-1} + \mu\nabla J(w_{i-1}) - \mu\nabla J(w^o)\|^2 \\ &= \|\widetilde{w}_{i-1}\|^2 + \mu^2 \underbrace{\|\nabla J(w_{i-1}) - \nabla J(w^o)\|^2}_{\text{Lipschitz gradients}} - 2\mu \underbrace{\widetilde{w}_{i-1}^{\mathsf{T}} \left(\nabla J(w^o) - \nabla J(w_{i-1})\right)}_{\text{strong convexity}} \\ &\leq \|\widetilde{w}_{i-1}\|^2 + \mu^2\delta^2\|\widetilde{w}_{i-1}\|^2 - 2\mu\nu\|\widetilde{w}_{i-1}\|^2 \\ &= \left(1 - 2\mu\nu + \mu^2\delta^2\right)\|\widetilde{w}_{i-1}\|^2 \end{aligned} \tag{9}$$

We have linear convergence whenever:

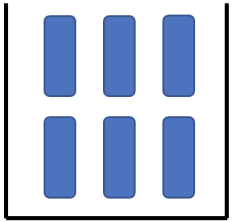
$$1 - 2\mu\nu + \mu^2\delta^2 < 1 \iff \mu < \frac{2\nu}{\delta^2} \tag{10}$$

Then:

$$\|\widetilde{w}_i\|^2 \leq \left(1 - 2\mu\nu + \mu^2\delta^2\right)^i \|\widetilde{w}_0\|^2 \tag{11}$$

From Optimization to Learning

- So far: no data and perfect knowledge of $J(w)$ and $\nabla J(w)$
- Learning problems: dependence on data and imperfect knowledge of $J(w)$ and $\nabla J(w)$
- $\boldsymbol{x}$: vector-valued random variable of unknown distribution representing data
- x_i: realization of $\boldsymbol{x}$ at time i



Inability of Gradient Descent to Minimize Risks

- $Q(w; \boldsymbol{x})$: the loss encountered by the model w on the data $\boldsymbol{x}$
- The loss is random! So look for the model w that minimizes the risk:

$$J(w) = \mathbb{E}Q(w; \boldsymbol{x}) \tag{12}$$

$$w^o = \arg\min_w J(w) \tag{13}$$

- Let's try gradient descent:

$$w_i = w_{i-1} - \mu \nabla J(w_{i-1}) = w_{i-1} - \mu \nabla \mathbb{E}Q(w_{i-1}; \boldsymbol{x}) \tag{14}$$

- We need $\nabla \mathbb{E}Q(w; \boldsymbol{x})$, but we don't know the probability density function of $\boldsymbol{x}$.

Empirical Risk Minimization

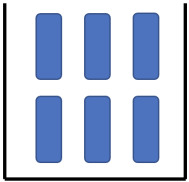
- Since we cannot minimize $\mathbb{E}Q(w;\boldsymbol{x})$ directly, collect realizations $\{x_n\}_{n=1}^N$ and approximate:

$$J^{\text{emp}}(w) = \frac{1}{N}\sum_{n=1}^{N} Q(w;x_n) \approx \mathbb{E}Q(w;\boldsymbol{x}) = J(w) \tag{15}$$

Caveat

Approximation is accuracte as $N \to \infty$ and for $\{x_n\}_{n=1}^N$ identically distributed.

- $J^{\text{emp}}(w)$ can be minimized via gradient descent, but is:
  - Computationally expensive
  - And fails if $\{x_n\}_{n=1}^N$ is not available at once, or not identically distributed.



Online Stochastic Gradient Descent

- Recal the (infeasible) gradient descent recursion:

$$w_i = w_{i-1} - \mu \nabla J(w_{i-1}) = w_{i-1} - \mu \nabla \mathbb{E} Q(\boldsymbol{w}_{i-1}; \boldsymbol{x}) \tag{16}$$

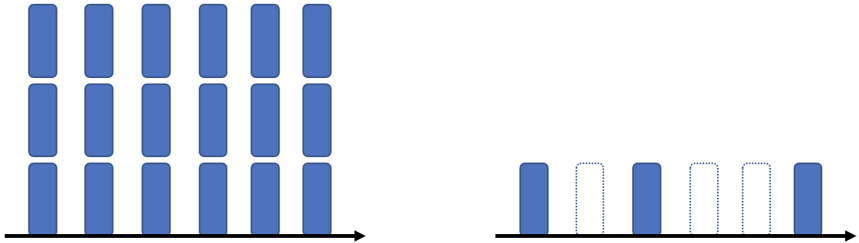
- Dropping the expected value, we obtain stochastic gradient descent:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu \nabla Q(\boldsymbol{w}_{i-1}; \boldsymbol{x}) \tag{17}$$

- $\boldsymbol{w}_i$ is now random (and bold), since $\nabla Q(\boldsymbol{w}_{i-1}; \boldsymbol{x})$ is random.

Variants of Stochastic Gradient Descent

- We might receive multiple samples at a time, or sometimes receive no samples at all.



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

$$\boldsymbol{w}_i = \begin{cases} \boldsymbol{w}_{i-1} - \frac{\mu}{p}\nabla Q(\boldsymbol{w}_{i-1};\boldsymbol{x}) & \text{with data} \\ \boldsymbol{w}_{i-1} & \text{without data.} \end{cases}$$

A Unifying Framework via Stochastic Gradient Approximations

- We study stochastic gradient descent with a generic gradient approximation:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu \widehat{\nabla J}(\boldsymbol{w}_{i-1}) \tag{18}$$

- Example #1: Ordinary SGD

$$\widehat{\nabla J}(\boldsymbol{w}_{i-1}) = \nabla Q(\boldsymbol{w}_{i-1}, \boldsymbol{x}) \tag{19}$$

- Example #2: Mini-Batch SGD

$$\widehat{\nabla J}(\boldsymbol{w}_{i-1}) = \frac{1}{B} \sum_{b=1}^{B} \nabla Q(\boldsymbol{w}_{i-1}, \boldsymbol{x}_b) \tag{20}$$

- Example #3: Asynchronous SGD

$$\widehat{\nabla J}(\boldsymbol{w}_{i-1}) = \begin{cases} \frac{1}{p} \nabla Q(\boldsymbol{w}_{i-1}; \boldsymbol{x}) & \text{with prob. } p, \\ 0 & \text{with prob. } 1-p. \end{cases} \tag{21}$$

- Example #4: Gradient Descent

$$\widehat{\nabla J}(\boldsymbol{w}_{i-1}) = \nabla J(\boldsymbol{w}_{i-1}) \tag{22}$$

Modeling Conditions

- Performance of SGD depends on the quality of $\widehat{\nabla J}(\boldsymbol{w}_{i-1})$
- We can write:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu\widehat{\nabla J}(\boldsymbol{w}_{i-1}) = \boldsymbol{w}_{i-1} - \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \tag{23}$$

- We introduced the gradient noise:

$$\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) = \widehat{\nabla J}(\boldsymbol{w}_{i-1}) - \nabla J(\boldsymbol{w}_{i-1}) \tag{24}$$

Gradient noise conditions

$$\mathbb{E}\left\{\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \middle| \boldsymbol{w}_{i-1}\right\} = 0 \tag{25}$$

$$\mathbb{E}\left\{\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \middle| \boldsymbol{w}_{i-1}\right\} \leq \beta^2\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \sigma^2 \tag{26}$$

Example: Least-squares (1)

Consider a regression model:

$$\boldsymbol{d} = \boldsymbol{u}^{\mathsf{T}} w^o + \boldsymbol{v} \tag{27}$$

We can find w^o via least-squares:

$$w^o = \arg\min_{w} \frac{1}{2} \mathbb{E} \left\| \boldsymbol{d} - \boldsymbol{u}^{\mathsf{T}} w^o \right\|^2 \tag{28}$$

We have:

$$\nabla J(w) = \left(\mathbb{E}\, \boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} \right) w - \mathbb{E}\, \boldsymbol{d}\boldsymbol{u} \tag{29}$$

$$\nabla Q(w; \boldsymbol{x}) = \left(\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} \right) w - \boldsymbol{d}\boldsymbol{u} \tag{30}$$

If we let $\widehat{\nabla J}(w) = \nabla Q(w; \boldsymbol{x})$, we hence have:

$$\boldsymbol{s}_i(w) = \left(\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} \right) w - \boldsymbol{d}\boldsymbol{u} - \left(\mathbb{E}\, \boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} \right) w - \mathbb{E}\, \boldsymbol{d}\boldsymbol{u} \tag{31}$$

Example: Least-squares (2)

Since $\boldsymbol{d} = \boldsymbol{u}^{\mathsf{T}} w^o + \boldsymbol{v}$, we can simplify:

$$\begin{aligned}
\boldsymbol{s}_i(w) &= \left(\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\right) w - \boldsymbol{d}\boldsymbol{u} - \left(\mathbb{E}\,\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\right) w - \mathbb{E}\,\boldsymbol{d}\boldsymbol{u} \\
&= \left(\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} - \mathbb{E}\,\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\right) w - (\boldsymbol{d}\boldsymbol{u} - \mathbb{E}\,\boldsymbol{d}\boldsymbol{u}) \\
&= \left(\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} - \mathbb{E}\,\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\right) w - \left(\left(\boldsymbol{u}^{\mathsf{T}} w^o + \boldsymbol{v}\right)\boldsymbol{u} - \mathbb{E}\left(\boldsymbol{u}^{\mathsf{T}} w^o + \boldsymbol{v}\right)\boldsymbol{u}\right) \\
&= \left(\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} - \mathbb{E}\,\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\right)(w - w^o) - (\boldsymbol{v}\boldsymbol{u} - \mathbb{E}\,\boldsymbol{v}\boldsymbol{u})
\end{aligned} \tag{32}$$

We can immediately verify that $\mathbb{E}\boldsymbol{s}_i(w) = 0$. For the variance, see next slide:

Example: Least-squares (2)

$$
\begin{aligned}
\mathbb{E}\,\|\boldsymbol{s}_i(w)\|^2 &= \mathbb{E}\left\|\left(\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} - \mathbb{E}\,\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\right)(w-w^o) - \boldsymbol{v}\boldsymbol{u}\right\|^2 \\
&= \mathbb{E}\left\|\left(\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} - \mathbb{E}\,\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\right)(w-w^o)\right\|^2 + \mathbb{E}\,\|\boldsymbol{v}\boldsymbol{u}\|^2 \\
&\quad + 2\,(\mathbb{E}\,\boldsymbol{v})\,\mathbb{E}\left(\boldsymbol{u}^{\mathsf{T}}\left(\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} - \mathbb{E}\,\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\right)(w-w^o)\right) \\
&\leq \underbrace{\mathbb{E}\left\|\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}} - \mathbb{E}\,\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\right\|^2}_{\beta^2}\underbrace{\|w-w^o\|^2}_{\|\widetilde{w}\|^2} + \underbrace{\mathbb{E}\,\boldsymbol{v}^2\,\mathbb{E}\,\|\boldsymbol{u}\|^2}_{\sigma^2}
\end{aligned}
\tag{33}
$$

Verifying Modeling Conditions

- Gradient noise conditions can be verified for a number of learning problems, e.g.:
  - Least-squares regression
  - Logistic regression
  - Certain deep learning formulations
- Parameters for variants of SGD can be deduced from those of ordinary SGD

$\widehat{\nabla J}(\boldsymbol{w}_{i-1})$	unbiased?	β^2 (relative)	σ^2 (absolute)
$\nabla Q(\boldsymbol{w}_{i-1}, \boldsymbol{x})$	yes	β^2_{ord}	σ^2_{ord}
$\frac{1}{B}\sum_{b=1}^{B} \nabla Q(\boldsymbol{w}_{i-1}, \boldsymbol{x}_b)$	yes	$\frac{\beta^2_{\text{ord}}}{B}$	$\frac{\sigma^2_{\text{ord}}}{B}$
$\nabla J(\boldsymbol{w}_{i-1})$	yes	0	0

Convergence of Stochastic Gradient Descent (1)

We extend the convergence proof of gradient descent to allow for stochastic approximations. Recall the generic stochastic gradient descent recursion:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu \nabla J(\boldsymbol{w}_{i-1}) - \mu\, \boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \tag{34}$$

Subtract from w^o to find the error recursion:

$$\widetilde{\boldsymbol{w}}_i = \widetilde{\boldsymbol{w}}_{i-1} + \mu \nabla J(\boldsymbol{w}_{i-1}) - \mu \underbrace{\nabla J(w^o)}_{=0} + \mu\, \boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \tag{35}$$

Convergence of Stochastic Gradient Descent (2)

Square and expand twice:

$$\begin{aligned}
\|\widetilde{\boldsymbol{w}}_i\|^2 &= \|\widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o) + \mu\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \\
&= \|\widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o)\|^2 + \mu^2\|\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \\
&\quad + 2\mu\,(\widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o))^{\mathsf{T}}\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \\
&= \left(1 - 2\mu\nu + \mu^2\delta^2\right)\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\|\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \\
&\quad + 2\mu\,(\widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o))^{\mathsf{T}}\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \qquad (36)
\end{aligned}$$

We take expectations, conditioned on $\boldsymbol{w}_{i-1}$, and obtain:

$$\begin{aligned}
\mathbb{E}\left\{\|\widetilde{\boldsymbol{w}}_i\|^2|\,\boldsymbol{w}_{i-1}\right\} &\le \left(1 - 2\mu\nu + \mu^2\delta^2\right)\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\,\mathbb{E}\left\{\|\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2|\,\boldsymbol{w}_{i-1}\right\} \\
&\quad + 2\mu\,(\widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o))^{\mathsf{T}}\,\mathbb{E}\left\{\boldsymbol{s}_i(\boldsymbol{w}_{i-1})|\,\boldsymbol{w}_{i-1}\right\} \\
&\le \left(1 - 2\mu\nu + \mu^2\delta^2\right)\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\beta^2\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\sigma^2 \\
&= \left(1 - 2\mu\nu + \mu^2\left(\delta^2 + \beta^2\right)\right)\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\sigma^2 \qquad (37)
\end{aligned}$$

Convergence of Stochastic Gradient Descent (3)

We take expectations again to remove the conditioning:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \left(1 - 2\mu\nu + \mu^2\left(\delta^2+\beta^2\right)\right)\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\sigma^2 \tag{38}$$

The recursion is stable whenever:

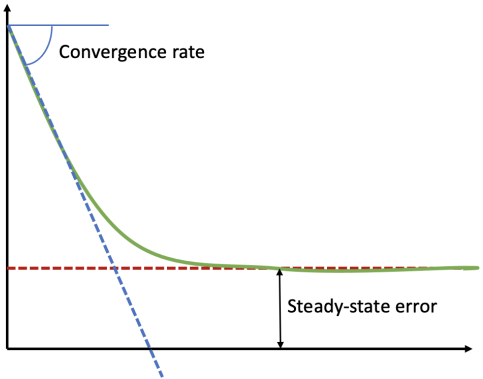
$$\mu < \frac{2\nu}{\delta^2+\beta^2} \tag{39}$$

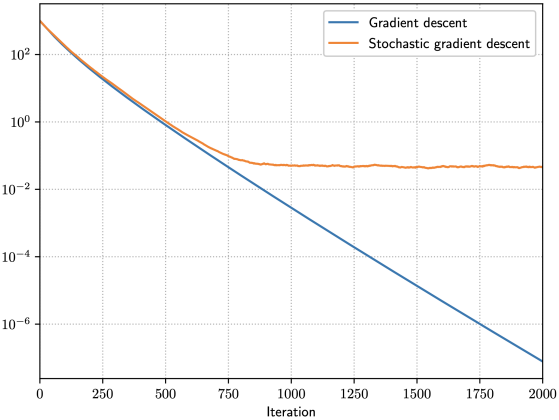
Upon iterating:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \left(1 - 2\mu\nu + \mu^2\left(\delta^2+\beta^2\right)\right)^i\|\widetilde{w}_0\|^2 + \frac{\mu\sigma^2}{2\nu - \mu(\beta^2+\delta^2)} \tag{40}$$

We observe linear convergence, but only up to $O(\mu\sigma^2)$.

Visualization

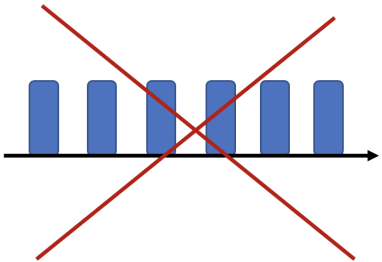


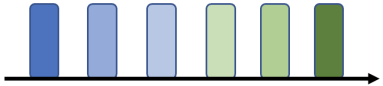


Part II: Dynamic Optimization

Static Optimization Problems

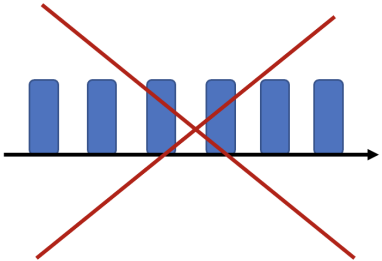
- We have seen SGD is able to perform online inference and optimization.
- But optimization problem and data have been static up to now:
  - Data $\boldsymbol{x}$, while random, is identically distributed over time
  - Risk $J(w) = \mathbb{E}\, Q(w; \boldsymbol{x})$, as a result, is constant over time
  - Hence, the minimizer w^o is also fixed

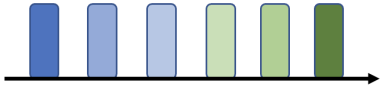




Dynamic Optimization Problems

- How will SGD perform when applied in dynamic environments?
- Everything becomes time-varying:
  - Data $\boldsymbol{x}_i$ is no longer distributed identically over time
  - Risk $J_i(w) = \mathbb{E}\, Q(w; \boldsymbol{x}_i)$, as a result, is time-varying
  - Hence, the minimizer w_i^o is also time-varying



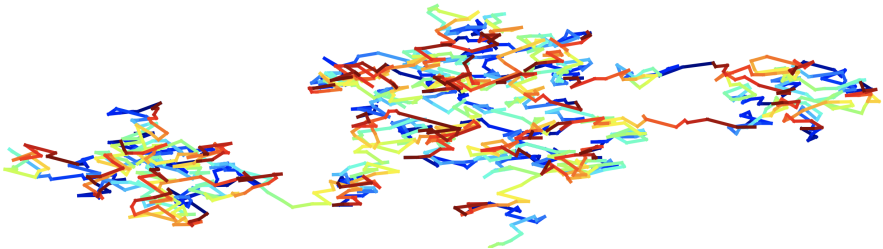


Random Walk Model

- Perhaps the most immediate model for time-varying costs imposes a random walk on the optimal models $\boldsymbol{w}_i^o$:

$$\boldsymbol{w}_i^o = \boldsymbol{w}_{i-1}^o + \boldsymbol{q}_i \tag{41}$$

- Motivation originates from adaptive filtering and target-tracking, where $\boldsymbol{w}_i^o$ represent coordinates of target



Bounded Deviations

- We assume:

$$\mathbb{E}\boldsymbol{q}_i = 0 \tag{42}$$

$$\mathbb{E}\|\boldsymbol{q}_i\|^2 \leq \sigma_q^2 \tag{43}$$

- Addtionally, we assume $\boldsymbol{q}_i$ independent of all other random variables (e.g., $\boldsymbol{x}_i$).
- Large σ_q^2 implies fast drift of $\boldsymbol{w}_i^o$.
- Bounds on the drift of $\boldsymbol{w}_i^o$ are sufficient to guarantee the tracking ability of SGD.

Strong Convexity and Lipschitz Gradients

We assume each $J_i(w)$ to be well-behaved:

- ν_i-strongly convex:

$$(\nabla J_i(x) - \nabla J_i(y))^{\mathsf{T}} (x - y) \geq \nu_i \|x - y\|^2 \iff \nabla^2 J_i(w) \geq \nu_i I_M \tag{44}$$

- δ_i-Lipschitz gradients:

$$\|\nabla J_i(x) - \nabla J_i(y)\| \leq \delta_i \|x - y\| \iff \nabla^2 J_i(w) \leq \delta_i I_M \tag{45}$$

- Assume some time-independent lower and upper bounds:

$$\nu_i \geq \nu \tag{46}$$

$$\delta_i \leq \delta \tag{47}$$

Modeling Conditions

- We can write:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu\widehat{\nabla J}_i(\boldsymbol{w}_{i-1}) = \boldsymbol{w}_{i-1} - \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \tag{48}$$

- We introduced the gradient noise:

$$\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) = \widehat{\nabla J}_i(\boldsymbol{w}_{i-1}) - \nabla J_i(\boldsymbol{w}_{i-1}) \tag{49}$$

Gradient noise conditions

$$\mathbb{E}\left\{\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\middle|\,\boldsymbol{w}_{i-1}\right\} = 0 \tag{50}$$

$$\mathbb{E}\left\{\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2\middle|\,\boldsymbol{w}_{i-1}\right\} \leq \beta_i^2\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \sigma_i^2 \tag{51}$$

with $\beta_i^2 \leq \beta^2$ and $\sigma_i^2 \leq \sigma^2$.

Tracking of Stochastic Gradient Descent (1)

We extend the convergence proof of stochastic gradient descent to allow for dynamic environments. Recall the generic stochastic gradient descent recursion:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu \nabla J(\boldsymbol{w}_{i-1}) - \mu\, \boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \tag{52}$$

Define $\widetilde{\boldsymbol{w}}_i = \boldsymbol{w}_i^o - w_i$. Subtract from $\boldsymbol{w}_i^o$ to find the error recursion:

$$\begin{aligned}
\widetilde{\boldsymbol{w}}_i &= \boldsymbol{w}_i^o - \boldsymbol{w}_{i-1} + \mu \nabla J(\boldsymbol{w}_{i-1}) - \mu \underbrace{\nabla J(w^o)}_{=0} + \mu\, \boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \\
&= \boldsymbol{w}_i^o - \boldsymbol{w}_{i-1}^o + \boldsymbol{w}_{i-1}^o - \boldsymbol{w}_{i-1} + \mu \nabla J(\boldsymbol{w}_{i-1}) - \mu \underbrace{\nabla J(w^o)}_{=0} + \mu\, \boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \\
&= \boldsymbol{q}_i + \widetilde{\boldsymbol{w}}_{i-1}^o + \mu \nabla J(\boldsymbol{w}_{i-1}) - \mu \underbrace{\nabla J(w^o)}_{=0} + \mu\, \boldsymbol{s}_i(\boldsymbol{w}_{i-1})
\end{aligned} \tag{53}$$

Tracking of Stochastic Gradient Descent (2)

Square and expand twice:

$$\begin{aligned}\|\widetilde{\boldsymbol{w}}_i\|^2 &= \|\boldsymbol{q}_i + \widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o) + \mu\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \\ &= \|\boldsymbol{q}_i\|^2 + \|\widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o) + \mu\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \\ &\quad + 2\left(\widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o) + \mu\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\right)^{\mathsf{T}}\boldsymbol{q}_i \end{aligned} \tag{54}$$

We take expectations, conditioned on $\boldsymbol{w}_{i-1}$, and obtain:

$$\begin{aligned}\mathbb{E}\left\{\|\widetilde{\boldsymbol{w}}_i\|^2 |\,\boldsymbol{w}_{i-1}\right\} &\le \mathbb{E}\left\{\|\boldsymbol{q}_i\|^2 |\,\boldsymbol{w}_{i-1}\right\} \\ &\quad + \mathbb{E}\left\{\|\widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o) + \mu\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 |\,\boldsymbol{w}_{i-1}\right\} \\ &\quad + 2\left(\widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\nabla J(w^o) + \mu\,\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\right)^{\mathsf{T}}\mathbb{E}\left\{\boldsymbol{q}_i |\,\boldsymbol{w}_{i-1}\right\} \\ &\le \sigma_q^2 + \left(1 - 2\mu\nu + \mu^2\left(\delta^2 + \beta^2\right)\right)\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\sigma^2 \end{aligned} \tag{55}$$

Tracking of Stochastic Gradient Descent (3)

We take expectations again to remove the conditioning:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \left(1 - 2\mu\nu + \mu^2\left(\delta^2 + \beta^2\right)\right)\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \sigma_q^2 + \mu^2\sigma^2 \tag{56}$$

The recursion is stable whenever:

$$\mu < \frac{2\nu}{\delta^2 + \beta^2} \tag{57}$$

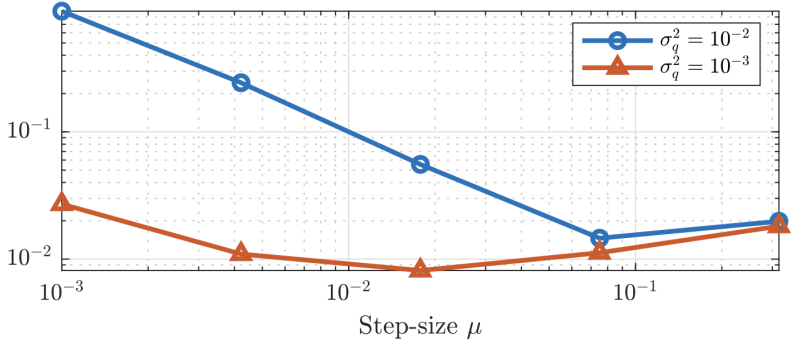
Upon iterating:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \underbrace{\left(1 - 2\mu\nu + \mu^2\left(\delta^2 + \beta^2\right)\right)^i \|\widetilde{w}_0\|^2}_{\text{transient error}} + \underbrace{\frac{\mu^{-1}\sigma_q^2}{2\nu - \mu(\beta^2 + \delta^2)}}_{\text{tracking error}} + \underbrace{\frac{\mu\sigma^2}{2\nu - \mu(\beta^2 + \delta^2)}}_{\text{steady-state error}} \tag{58}$$

We observe linear convergence, but only up to $O(\mu\sigma^2) + O(\mu^{-1}\sigma_q^2)$.

Trade-offs of Dynamic Online Learning

$$\limsup_{i\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i\|^2 = \underbrace{\frac{\mu^{-1}\sigma_q^2}{2\nu-\mu(\beta^2+\delta^2)}}_{\text{tracking error}} + \underbrace{\frac{\mu\sigma^2}{2\nu-\mu(\beta^2+\delta^2)}}_{\text{steady-state error}} \tag{59}$$



Part III: Extensions and Applications

Multi-Agent Systems

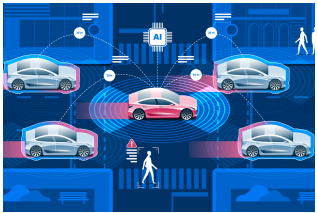


Figure: Autonomous vehicles [smartcitiesworld.net]

Figure: Social network [medium.com]

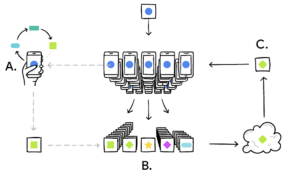


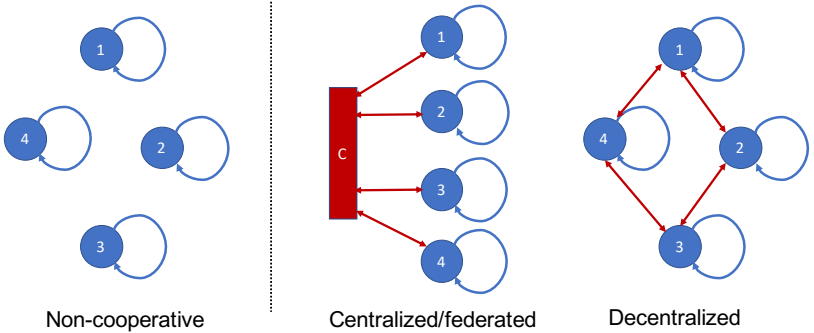
Figure: Mobile devices [ai.googleblog.com]

Figure: Drone swarms [ft.com]

Learning over Graphs

- Given a collection of K "agents", indexed by k, we seek globally optimal behavior:

$$w^o \triangleq \min_w \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{\boldsymbol{x}_k} Q(w; \boldsymbol{x}_k)$$



Federated Averaging (1)

- We examine the Federated Averaging Algorithm [McMahan et al 2017].
- At time i, server picks B agents from the index set $\{1, \ldots, K\}$ and collects them in the set $\mathcal{B}$. Broadcasts $\boldsymbol{w}^{i-1}$ to them.
- Each picked agent takes the model and performs a local gradient update:

$$\boldsymbol{w}_b^+ = \boldsymbol{w}^{i-1} - \mu \nabla Q(\boldsymbol{w}_{i-1}; \boldsymbol{x}_b) \tag{60}$$

- The server aggregates:

$$\boldsymbol{w}_{i-1} = \frac{1}{B} \sum_{b \in \mathcal{B}} \boldsymbol{w}_b^+ \tag{61}$$

Federated Averaging (2)

- If we combine (60) and (61), we find:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \frac{\mu}{B}\sum_{b\in\mathcal{B}} \nabla Q(\boldsymbol{w}_{i-1}, \boldsymbol{x}_b) \tag{62}$$

- This relation is very reminiscent of the mini-batch SGD recursion. Indeed, if we let:

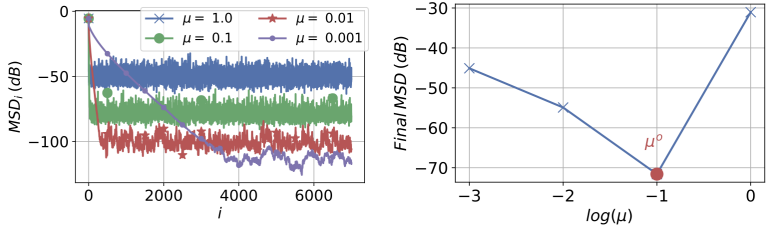
$$\widehat{\nabla J}(\boldsymbol{w}_{i-1}) \triangleq \frac{1}{B}\sum_{b\in\mathcal{B}} \nabla Q(\boldsymbol{w}_{i-1}; \boldsymbol{x}_b) \tag{63}$$

- We can verify that $\widehat{\nabla J}(\boldsymbol{w}_{i-1})$ satisfies the gradient noise bounds from Part II, and Federated Averaging is equivalent to:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu\widehat{\nabla J}(\boldsymbol{w}_{i-1}) = \boldsymbol{w}_{i-1} - \mu\nabla J(\boldsymbol{w}_{i-1}) - \mu\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \tag{64}$$

- All results from Part II apply!

Federated Averaging (3)

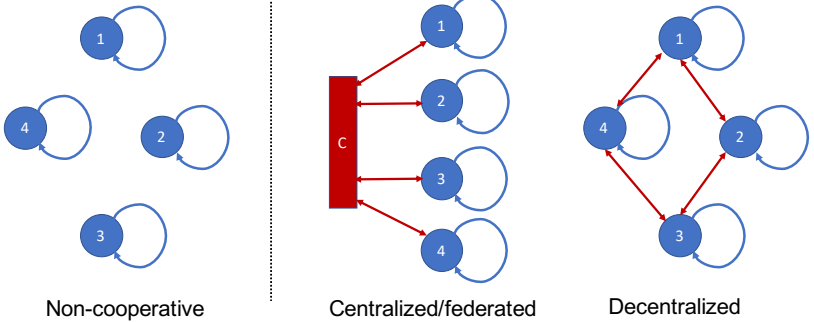


(a) *Stationary case:* varying μ (b) *Non-stationary case:* varying μ

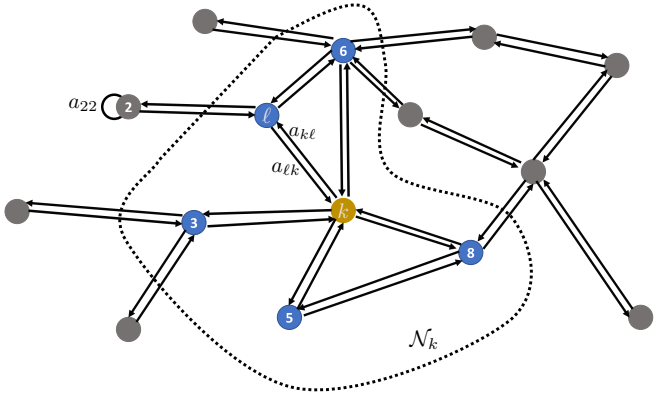
- "Dynamic Federated Learning", Elsa Rizk, Stefan Vlaski and Ali H. Sayed, 2020

Decentralized Optimization and Learning

- A fusion center can cause concerns around:
  - Privacy
  - Robustness
  - Communication efficiency



Network Model



- Graph of K agents with weighted, doubly-stochastic, adjacency matrix A and $a_{k\ell} = [A]_{k\ell} \geq 0$ if there is an edge connecting k and ℓ.

Deriving Decentralized Algorithms

- There is a plethora of decentralized algorithms. Rather than be exhaustive, we will show two families, and approaches to derive them:
  - Primal decentralized algorithms
  - Primal-dual decentralized algorithms
- In both cases, we make the following reformulation:

$$\begin{aligned} & \min_{w} \sum_{k=1}^{K} J_k(w) \\ \iff & \min_{w_k} \sum_{k=1}^{K} J_k(w_k) \text{ subject to } w_k = w_\ell \ \forall \ k, \ell \\ \iff & \min_{w_k} \sum_{k=1}^{K} J_k(w_k) \text{ subject to } w_k = w_\ell \ \forall \ \ell \in \mathcal{N}_k \end{aligned} \tag{65}$$

Primal Decentralized Algorithms (1)

It is useful to define "network quantities":

$$\mathcal{W} \triangleq \operatorname{col}\{w_1, w_2, \ldots, w_K\} \in \mathbb{R}^{MK} \tag{66}$$

$$\mathcal{J}(\mathcal{W}) \triangleq \sum_{k=1}^{K} J_k(w_k) \tag{67}$$

We can then write the global objective more compactly as:

$$\min_{\mathcal{W}} \mathcal{J}(\mathcal{W}) + \frac{\eta}{2} \sum_{k=1}^{K} \sum_{\ell \in \mathcal{N}_k} c_{\ell k} \|w_k - w_\ell\|^2 \tag{68}$$

Primal Decentralized Algorithms (2)

For the penalty term:

$$\frac{\eta}{2}\sum_{k=1}^{K}\sum_{\ell\in\mathcal{N}_k} c_{\ell k}\|w_k - w_\ell\|^2 = \frac{\eta}{2}\,\mathcal{W}^{\mathsf{T}}\,\mathcal{L}\,\mathcal{W} \tag{69}$$

where $\mathcal{L} = L \otimes I_M$ and we defined:

$$L = \operatorname{diag}\{C\mathbb{1}\} - C \tag{70}$$

Under those definitions we find:

$$\min_{\mathcal{W}} \mathcal{J}(\mathcal{W}) + \frac{\eta}{2}\,\mathcal{W}^{\mathsf{T}}\,\mathcal{L}\,\mathcal{W} \tag{71}$$

For brevity, we shall define:

$$\mathcal{J}^{\eta}(\mathcal{W}) \triangleq \mathcal{J}(\mathcal{W}) + \frac{\eta}{2}\,\mathcal{W}^{\mathsf{T}}\,\mathcal{L}\,\mathcal{W} \tag{72}$$

Primal Decentralized Algorithms (3)

Applying gradient descent to (71), we obtain the recursion:

$$\begin{aligned}\mathcal{W}_i &= \mathcal{W}_{i-1} - \mu\left(\nabla_{\mathcal{W}}\mathcal{J}(\mathcal{W}_{i-1}) + \eta\mathcal{L}\,\mathcal{W}_{i-1}\right) \\ &= (I - \mu\eta\mathcal{L})\,\mathcal{W}_{i-1} - \mu\nabla_{\mathcal{W}}\mathcal{J}(\mathcal{W}_{i-1})\end{aligned} \tag{73}$$

For simplicity, it is common to couple the regularization parameter η to the step-size μ and let $\eta = \mu^{-1}$, resulting in:

$$\mathcal{W}_i = \mathcal{A}^{\mathsf{T}}\,\mathcal{W}_{i-1} - \mu\nabla_{\mathcal{W}}\mathcal{J}(\mathcal{W}_{i-1}) \tag{74}$$

where we defined $\mathcal{A} = A \otimes I_M$ with:

$$A \triangleq I - L \tag{75}$$

Primal Decentralized Algorithms (4)

Exploiting the block-structure:

$$\nabla_{\mathcal{W}} \mathcal{J}(\mathcal{W}_{i-1}) = \begin{pmatrix} \nabla_{w_1} J_1(w_{1,i-1}) \\ \nabla_{w_2} J_2(w_{2,i-1}) \\ \vdots \\ \nabla_{w_K} J_K(w_{K,i-1}) \end{pmatrix} \tag{76}$$

We can then expand (74):

$$\begin{pmatrix} w_{1,i-1} \\ w_{2,i-1} \\ \vdots \\ w_{K,i-1} \end{pmatrix} = \mathcal{A}^{\mathsf{T}} \begin{pmatrix} w_{1,i-1} \\ w_{2,i-1} \\ \vdots \\ w_{K,i-1} \end{pmatrix} - \mu \begin{pmatrix} \nabla_{w_1} J_1(w_{1,i-1}) \\ \nabla_{w_2} J_2(w_{2,i-1}) \\ \vdots \\ \nabla_{w_K} J_K(w_{K,i-1}) \end{pmatrix} \tag{77}$$

Since the recursion has now been fully decoupled into blocks:

$$w_{k,i} = \sum_{\ell=1}^{K} a_{\ell k} w_{\ell,i-1} - \mu \nabla_{w_k} J_k(w_{k,i-1}) \tag{78}$$

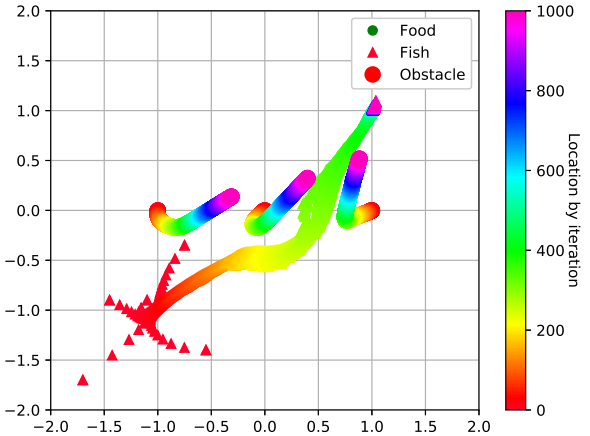
Decentralized (Stochastic) Gradient Descent

- We have just derived the Decentralized (Stochastic) Gradient Descent Algorithm, also known as the consensus algorithm [Nedic and Ozdaglar 2009].
- Gradient-based, so can be online and track drifts!

$$\boldsymbol{w}_{k,i} = \sum_{\ell=1}^{K} a_{\ell k}\, \boldsymbol{w}_{\ell,i-1} - \mu \widehat{\nabla J}_{k,i}(\boldsymbol{w}_{k,i-1}) \tag{79}$$

- Proving tracking ability is a small variation of Part II. For details, see:
  - "On distributed online classification in the midst of concept drifts", Zaid Towfic, Jianshu Chen, Ali H. Sayed, 2013
  - "Tracking Performance of Online Stochastic Learners", Stefan Vlaski, Elsa Rizk, Ali H. Sayed, 2020
- Drawback: Based on penalizing constraints, so has a (small) bias.

Example: Avoiding Moving Predators



Primal-Dual Decentralized Algorithms (1)

Rather than penalize, we now insist on perfect consensus:

$$\min_{w_k} \sum_{k=1}^{K} J_k(w_k) \text{ subject to } \frac{1}{2} \sum_{k=1}^{K} \sum_{\ell \in \mathcal{N}_k} c_{\ell k} \|w_k - w_\ell\|^2 = 0 \tag{80}$$

More compactly:

$$\min_{\mathcal{W}} \mathcal{J}(\mathcal{W}) \text{ subject to } \frac{1}{2} \mathcal{W}^{\mathsf{T}} \mathcal{L} \mathcal{W} = 0 \tag{81}$$

Since $L = L^{\mathsf{T}}$ is symmetric and positive semi-definite, it admits a Cholesky decomposition of the form:

$$L = B^{\mathsf{T}} B \tag{82}$$

We then have:

$$0 = \frac{1}{2} \mathcal{W}^{\mathsf{T}} \mathcal{L} \mathcal{W} = \frac{1}{2} \mathcal{W}^{\mathsf{T}} \mathcal{B}^{\mathsf{T}} \mathcal{B} \mathcal{W} = \frac{1}{2} \|\mathcal{B} \mathcal{W}\|^2 \iff \mathcal{B} \mathcal{W} = 0 \tag{83}$$

Primal-Dual Decentralized Algorithms (2)

Hence, an equivalent problem is:

$$\min_{\mathcal{W}} \mathcal{J}(\mathcal{W}) \text{ subject to } \mathcal{B}\,\mathcal{W} = 0 \tag{84}$$

The augmented Lagrangian for problem (84) corresponds to:

$$\mathcal{L}(\mathcal{W}, \lambda) = \mathcal{J}(\mathcal{W}) + \lambda^{\mathsf{T}}\mathcal{B}\,\mathcal{W} + \frac{\eta}{2}\|\mathcal{B}\,\mathcal{W}\|^2 = \mathcal{J}(\mathcal{W}) + \lambda^{\mathsf{T}}\mathcal{B}\,\mathcal{W} + \frac{\eta}{2}\,\mathcal{W}^{\mathsf{T}}\,\mathcal{L}\,\mathcal{W} \tag{85}$$

We can then alternately perform a gradient *descent* step on the primal variable $\mathcal{W}$ along with a gradient *ascent* step on the dual variable λ. The resulting algorithm amounts to:

$$\begin{aligned}
\mathcal{W}_i &= \mathcal{W}_{i-1} - \mu\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \mu\mathcal{B}^{\mathsf{T}}\lambda_{i-1} - \mu\eta\mathcal{L}\,\mathcal{W}_{i-1} \\
&= (I - \mu\eta\mathcal{L})\,\mathcal{W}_{i-1} - \mu\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \mu\mathcal{B}^{\mathsf{T}}\lambda_{i-1} && (86) \\
&= \mathcal{A}^{\mathsf{T}}\,\mathcal{W}_{i-1} - \mu\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \mu\mathcal{B}^{\mathsf{T}}\lambda_{i-1} && (87) \\
\lambda_i &= \lambda_{i-1} + \mu\mathcal{B}\,\mathcal{W}_{i-1} && (88)
\end{aligned}$$

Primal-Dual Decentralized Algorithms (3)

Propagation of λ_i is cumbersome. For the iteration from time $i-2$ to time $i-1$, we have from (86):

$$\mathcal{W}_{i-1} = \mathcal{A}^{\mathsf{T}}\,\mathcal{W}_{i-2} - \mu\nabla\mathcal{J}(\mathcal{W}_{i-2}) - \mu\mathcal{B}^{\mathsf{T}}\lambda_{i-2} \tag{89}$$

Subtracting (89) from (86), we find:

$$\begin{aligned}
\mathcal{W}_i - \mathcal{W}_{i-1} &= \mathcal{A}^{\mathsf{T}}\,(\mathcal{W}_{i-1} - \mathcal{W}_{i-2}) - \mu\,(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})) - \mu\mathcal{B}^{\mathsf{T}}\,(\lambda_{i-1} - \lambda_{i-2}) \\
&\overset{(88)}{=} \mathcal{A}^{\mathsf{T}}\,(\mathcal{W}_{i-1} - \mathcal{W}_{i-2}) - \mu\,(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})) - \mu\mathcal{B}^{\mathsf{T}}\mathcal{B}\,\mathcal{W}_{i-2} \\
&= (I - \mu\eta\mathcal{L})\,(\mathcal{W}_{i-1} - \mathcal{W}_{i-2}) - \mu\,(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})) - \mu\mathcal{L}\,\mathcal{W}_{i-2} \\
&= (I - \mu\eta\mathcal{L})\,\mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\,(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})) \\
&= \mathcal{A}^{\mathsf{T}}\,\mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\,(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2}))
\end{aligned} \tag{90}$$

It can be insightful to decompose (90) into two subsequent steps:

$$\phi_i = \mathcal{A}^{\mathsf{T}}\,\mathcal{W}_{i-1} - \mu\nabla\mathcal{J}(\mathcal{W}_{i-1}) \tag{91}$$

$$\mathcal{W}_i = \phi_i + \mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\mathcal{J}(\mathcal{W}_{i-1}) \tag{92}$$

Primal-Dual Decentralized Algorithms (4)

- We just derived the EXTRA algorithm [Wei Shi et al 2014]. It also decomposes, and can be run online with gradient approximations and tracking ability:

$$\boldsymbol{w}_{k,i} = \sum_{\ell=1}^{K} a_{\ell k}\, \boldsymbol{w}_{\ell,i-1} - \boldsymbol{w}_{k,i-2} - \mu \left(\widehat{\nabla J}_{k,i}(\boldsymbol{w}_{k,i-1}) - \widehat{\nabla J}_{k,i-1}(\boldsymbol{w}_{k,i-2}) \right) \tag{93}$$

- Again, to study tracking ability, we only require small variations to the framework of Part II. For details, see:
  - "Can Primal Methods Outperform Primal-dual Methods in Decentralized Dynamic Optimization?", Kun Yuan, Wei Xu, Qing Ling, 2020.

Sketch of Analysis

- The central inequality for establishing the tracking ability of SGD in Part II was:

$$\mathbb{E}\,\|\,\boldsymbol{w}_i^o - \boldsymbol{w}_i\|^2 \leq \left(1 - 2\mu\nu + \mu^2\left(\delta^2 + \beta^2\right)\right)\mathbb{E}\,\|\,\boldsymbol{w}_{i-1}^o - \boldsymbol{w}_{i-1}\|^2 + \mu^2\sigma^2 + \sigma_q^2 \tag{94}$$

- Mean-square-contractive: A general property of many mappings
- Decentralized algorithms are als mean-square contractive, albeit with different constants:

$$\mathbb{E}\,\|\,\boldsymbol{\mathcal{W}}_i^\infty - \boldsymbol{\mathcal{W}}_i\|^2 \leq \Gamma\,\mathbb{E}\,\|\,\boldsymbol{\mathcal{W}}_{i-1}^\infty - \boldsymbol{\mathcal{W}}_{i-1}\|^2 + \Sigma \tag{95}$$

Take-away

Proofs for all mean-square-contractive mappings are the same after accounting for $\boldsymbol{\mathcal{W}}_i^\infty, \Gamma, \Sigma$.

- For details, see "Tracking Performance of Online Stochastic Learners", Stefan Vlaski, Elsa Rizk, Ali H. Sayed, 2020.

Conclusion

- The state of nature can change, so our learning algorithms should respond
- We took a series of small steps:
  - From gradient descent to online stochastic gradient descent via gradient approximations and noise
  - From stochastic gradient descent to tracking drifts via random walk models
  - From stochastic gradient descent to distributed algorithms via mean-square contractive mappings
- Take-away: SGD-based algorithms have a provable ability to track drifts with quantifiable accuracy
- What's next: What if the evolution of $J_i(w)$ is no longer independent of our estimate $\boldsymbol{w}_i$?