Distributed Asynchronous Stochastic Approximation Algorithms with unbounded Stochastic Information Delays – Theory and Applications

Adrian Redder

Dissertation

in partial fulfillment of the requirements for the degree of
Doctor rerum naturalium (Dr. rer. nat.)

submitted to

Faculty of Computer Science, Electrical Engineering and Mathematics
Paderborn University, Germany

Advisors:
Prof. Dr. Holger Karl
Prof. Dr. Arunselvan Ramaswamy

Paderborn, January 2024

Dissertation

Distributed Asynchronous Stochastic Approximation Algorithms with unbounded Stochastic Information Delays – Theory and Applications.

Referees:

Prof. Dr. Holger Karl, Hasso Plattner Institute, University of Potsdam, Germany

Prof. Dr. Arunselvan Ramaswamy, Karlstad University, Sweden

Faculty of Computer Science, Electrical Engineering and Mathematics
Department of Computer Science
Paderborn University

To

My Parents & Eva

Acknowledgement

First and foremost, I want to thank my two advisors, Prof. Dr. Holger Karl and Prof. Dr. Arunselvan Ramaswamy, for their valuable guidance, critique, and patience throughout my Ph.D. and academic career. Further, I want to thank the SFB901 for their financial support. Finally, I want to thank my friends, family, parents, and especially Eva for their invaluable support, time, concern, and love throughout my life and academic career.

Abstract

At the advent of 2024, artificial intelligence (AI)-driven language and robotic systems are revolutionizing various domains and are expected to push the boundaries of human capabilities. The successes are based on advanced algorithms, but most importantly, on a growing consumption of computing resources. Training models with limited resources is therefore as important for reducing training time as it is for pushing the size of large AI models to new limits. This requires understanding the stability and convergence of learning algorithms, the most efficient use of parallel computing infrastructure, and how to adapt to errors caused by asynchronous, lightweight implementations. To advance our understanding of these problems, the present thesis studies distributed stochastic approximation (SA) algorithms. Such algorithms are defined by a coupled system of iterations that are adapted asynchronously by updates computed by a potentially large number of parallel computing resources. The framework jointly covers both asynchronous training of AI models as well as multi-agent learning in physically decentralized systems. The property that classifies these two scenarios as theoretically equivalent is that a set of variables is updated as a function of old versions of itself. In other words, the resulting iterations are affected by Age-of-Information (AoI).

Part I of this thesis studies the stability and convergence of distributed SA algorithms affected by AoI. The main result is a generalization of Borkar and Meyn's SA stability theorem to distributed SA algorithms, previously known only for bounded AoI processes. Beyond distributed SA, this result enables a novel stability theorem for SA algorithms with heavy ball momentum. As an application of the established distributed BMT, the thesis studies asynchronous stochastic gradient descent, for which an AoI-dependent convergence rate estimate is presented. Finally, a special distributed SA setting for Markov games is considered. First, a novel deep multi-agent actor-critic (AC) reinforcement learning algorithm is proposed, which is suitable for online learning based on communicated data. Second, the algorithm's convergence is obtained under mild communication assumptions. Furthermore, conditions on the AoI of data sampled from the environment (the controlled Markov process) are characterized to ensure convergence of the resulting state Markov process to a stationary distribution.

The fundamental property for the distributed SA analyses in Part I is the existence of random

variables with certain moment bounds that stochastically dominate the respective AoI processes. Part II of this thesis thus studies different AoI models to characterize the existence of AoI dominating random variables and, in general, AoI distributions. First, core growth properties that define AoI as a stochastic process are defined, and different source processes to describe AoI in different scenarios are identified. In particular, it is shown how strongly mixing event processes give rise to AoI processes. This versatile model can represent various AoI processes, e.g., AoI induced by point processes with dependent service times or AoI processes due to interference in wireless networks. Finally, an in-depth weak convergence analysis of AoI arising from asynchronous computing modeled as a parallel point process is discussed.

Publications and preprints based on this thesis

1) Redder, A., Ramaswamy, A., Karl, H. (2022). Multi-agent Policy Gradient Algorithms for Cyber-physical Systems with Lossy Communication. 14th International Conference on Agents and Artificial Intelligence (pp. 282-289).
2) Redder, A., Ramaswamy, A., Karl, H. (2022). Practical network conditions for the convergence of distributed optimization. 9th IFAC Conference on Networked Systems, 55(13), 133-138.
3) Redder, A., Ramaswamy, A., Karl, H. (2022). Age of Information Process under Strongly Mixing Communication-Moment Bound, Mixing Rate and Strong Law. In 2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton) (pp. 1-8). IEEE.
4) Redder, A., Ramaswamy, A., Karl, H. (2023). Stability and Convergence of Distributed Stochastic Approximations with large Unbounded Stochastic Information Delays. (Under review) arXiv preprint arXiv:2305.07091.
5) Redder. A. (2023) Age-of-Information in Distributed Systems caused by Asynchronous Computing Modeled as Parallel Renewal Processes, (Under review Statistics and Computing - Springer) https://doi.org/10.21203/rs.3.rs-3502945/v2.
6) Redder, A., Ramaswamy, A., Karl, H. (2022). Asymptotic convergence of deep multi-agent actor-critic algorithms. arXiv:2201.00570v1.
7) Redder, A., Ramaswamy, A., Karl, H. (2022). 3DPG: Distributed deep deterministic policy gradient algorithms for networked multi-agent systems. arXiv preprint arXiv:2201.00570v2.

Contents

Notation, Definitions & Background

Notation

- The natural numbers and the natural numbers with zero are denoted by $\mathbb{N}$ and $\mathbb{N}_0$, respectively. The whole numbers are denoted by $\mathbb{Z}$.
- The d-dimensional real coordinate space is denoted by $\mathbb{R}^d$ for all $d \in \mathbb{N}$.
- If not otherwise stated, $\|\cdot\|$ denotes some norm on a real coordinate space.
- Throughout, we reserve n to denote discrete time and we refer with $n \geq m$ to some $n \in \{n \in \mathbb{N}_0 : n \geq m\}$ for every $m \in \mathbb{N}_0$.
- Throughout, lowercase letters x will be used for deterministic objects or realizations of random objects, uppercase letters X will be used for random objects, and aside from the aforementioned number systems, calligraphic uppercase letters $\mathcal{X}$ refer to sets.
- The set of continuous functions from $\mathbb{R}^d$ to $\mathbb{R}^d$ is denoted by $\mathcal{C}(\mathbb{R}^d)$.
- A sequence of real vectors $x_0, x_1, \ldots, x_n, \ldots$ in $\mathbb{R}^d$ for some $d \in \mathbb{N}$, is denoted by $\{x_n\}_{n\geq 0}$ and will be abbreviated by $\{x_n\}$.
- Small o and big $\mathcal{O}$ notation: Consider two real-valued sequences x_n, y_n. Then, $x_n \in o(y_n)$ if $\limsup_{n\to\infty} \frac{x_n}{y_n} = 0$ and $x_n \in \mathcal{O}(y_n)$ if $\limsup_{n\to\infty} \frac{x_n}{y_n} < \infty$.
- Floor and ceiling function: For some $x \in \mathbb{R}$, the floor function is defined as $\lfloor x \rfloor := \max\{n \in \mathbb{Z} : n \leq x\}$ and the ceiling function is defined as $\lceil x \rceil := \min\{n \in \mathbb{Z} : n \geq x\}$.

Real Analysis and Dynamical Systems

For background on real analysis, we refer to Rudin (1987). For a sharp background on ordinary differential equations (ODEs), we refer to V. Borkar (2022, App. B). Here, we recall some terminology.

Definition 0.0.1 (Lipschitz continuous)**.** *A map $h : \mathbb{R}^n \to \mathbb{R}^m$ is called Lipschitz continuous, if $\|h(x) - h(y)\| \leq \|x - y\|$ for every $x, y \in \mathbb{R}^n$.*

Definition 0.0.2 (Locally Lipschitz continuous)**.** *A map $h : \mathbb{R}^n \to \mathbb{R}^m$ is called locally Lipschitz continuous if it is Lipschitz continuous when restricted to a compact set.*

Definition 0.0.3 (Invariant set). *For an ODE $\dot{x}(t) = h(x(t))$ in $\mathbb{R}^d$, a set $\mathcal{X} \subset \mathbb{R}^d$ is called invariant, if $x(0) \in \mathcal{X}$ implies that $x(t) \in \mathcal{X}$ for all $t \in \mathbb{R}$.*

Definition 0.0.4 (Equilibrium). *For an ODE $\dot{x}(t) = h(x(t))$ in $\mathbb{R}^d$, a point $x^* \in \mathbb{R}^d$ is called an equilibrium point, if $h(x^*) = 0$.*

Definition 0.0.5 (Global asymptotic stability). *An equilibrium point $x^* \in \mathbb{R}^d$ of an ODE is called globally asymptotically stable if all trajectories of the ODE converge to x^*.*

Probability Theory

We will use measure theoretic probability theory (Pollard 2002) and renewal theory (Cox and Isham 1980) tools throughout this work. For every considered setting, we always assume an underlying sufficiently rich probability space $(\Omega, \mathcal{F}, \mathbb{P})$ by Kolmogorov's Existence Theorem (Billingsley 2008, p. 482). All events are to be understood as elements of $\mathcal{F}$, and all random variables are measurable functions from Ω to another measurable space; all random variables in this work are either real- or integer-valued. We use {statement} as a short notation for $\{\omega \in \Omega : \text{statement is true}\}$, e.g. for a random variable X, $\{X = c\}$ denotes $\{\omega \in \Omega : X = c\}$. For a sequence of events $\{A(n)\}$, $\limsup A(n) \coloneqq \{A(n) \text{ i.o.}\}$ is the set of outcomes $\omega \in \Omega$ that occur infinitely often in $\{A(n)\}$. With a slight "abuse" of notation, we interchangeably refer with $A \in \mathcal{F}$ to the event A itself and to its corresponding indicator function $\mathbb{1}_A$. The abuse of notation follows the de Finetti notation (De Finetti 1970) as popularized by David Pollard. For a real-valued random variable X, its expected value is defined as the Lebesgue integral, $\mathbb{E}[X] \coloneqq \int_\Omega X d\mathbb{P}$.

Definition 0.0.6 (First order stochastic dominance). *A random variable X is said to be (first order)* stochastically dominated *by a random variable $\overline{X}$, denoted by $X \leq_{st} \overline{X}$, if*

$$\mathbb{P}(X > m) \leq \mathbb{P}\left(\overline{X} > m\right)$$

for all $m \geq 0$.

Definition 0.0.7 (Almost sure convergence). *A sequence of random variables $X(n)$ is said to converge almost surely (a.s.) to a random variable X, if $\mathbb{P}\left(\lim_{n\to\infty} X(n) = X\right) = 1$.*

Definition 0.0.8 (Weak convergence). *A sequence of real-valued random variables $X(n)$ is said to converge weakly (in distribution) to a random variable X if $\lim_{n\to\infty} \mathbb{P}(X(n) \leq x) = \mathbb{P}(X \leq x)$ for all $x \in \mathbb{R}$ where $\mathbb{P}(X \leq x)$ is continuous.*

We denote that $X(n)$ converges to X weakly by $X(n) \Rightarrow X$. Further, $X \sim Y$ denotes that X and Y are equal in distribution.

Definition 0.0.9 (Martingale & martingale difference sequence)**.** *Let $\{\mathcal{F}_n\}$ be an increasing family of sub σ-algebras of $\mathcal{F}$, then a sequence of real-valued random variables $\{X(n)\}$ is said to be* martingale*, if it is integrable with*

1) X_n is $\mathcal{F}_n$-measurable for all n.
2) $\mathbb{E}[X_{n+1} \mid \mathcal{F}_n] = X_n$ a.s. for all n.

In this case, the sequence $M_n := X_n - X_{n-1}$ is called a martingale difference sequence.

Renewal Theory

An ordinary (or zero delayed) renewal process on the real line is defined by a sequence of independent identically distributed (i.i.d.) interarrival times $\{W(n)\}$, such that

$$S(0) := 0,\ S(n) := S(n-1) + W(n), n \geq 1, \tag{1}$$

are the time steps where renewals occur (Cox 1962). Let F be the distribution function of the interarrival times, i.e., $F(x) = \mathbb{P}(W(n) \leq x)$ for all $x \geq 0$. The distribution function of $S(n)$ is given by the n-fold convolution of F: $\mathbb{P}(S(n) \leq x) =: F^{n*}(x)$. In this thesis, we consider $F(0) = 0$, i.e., renewal processes without multiple simultaneous occurrences.

Definition 0.0.10 (Renewal process)**.** *The* renewal process *associated with i.i.d. interarrival times $\{W(n)\}$ is*

$$N(t) := \max\{n : S(n) \leq t\}. \tag{2}$$

The renewal process is called lattice (more precisely, 1-lattice) if $W(n) \in \mathbb{N}$ for all $n \geq 1$.

A renewal process is called stationary if it has stationary increments. This property holds asymptotically for every renewal process. Further, a renewal process can be started in stationary mode by setting a different initial interarrival time.

Definition 0.0.11 (Modified renewal process)**.** *Let $N(t)$ be a renewal process with interarrival distribution F and $\mu := \mathbb{E}[W(n)] < \infty$, then the associated modified renewal process $\tilde{N}(t)$ is defined by replacing the first interarrival time $W(1)$ of $N(t)$ by $\tilde{W}(1)$ with distribution function*

$$G(x) := \mathbb{P}\left(\tilde{W}(1) \leq x\right) = \frac{1}{\mu}\int_0^x (1 - F(y))dy. \tag{3}$$

As defined in Definition 0.0.11, a modified renewal process is stationary increments (Serfozo 2009, Thm. 76).

Definition 0.0.12 (Backward recurrence time)**.** *Let $N(t)$ be a renewal process, then*

$$B(t) := t - S(N(t))$$

is called the backward recurrence time at time t, which is the time since the last *renewal before time t.*

Chapter 1

Introduction

The fields of artificial intelligence (AI), computing and optimization have made significant progress in recent years. Lately, large language models (LLMs) have revolutionized the natural language AI world since the seminal work of Vaswani et al. (2017). The seeds of artificial general intelligence are arguably visible (Bubeck et al. 2023), and robotic AI-driven autonomous systems, augmented with generative AI technologies such as LLM (Tagliabue et al. 2023), will revolutionize various domains in the coming decades by solving complex problems in a resilient manner (Kunze et al. 2018). Multi-agent robotic systems will further push the boundaries of human capabilities, by leveraging autonomy and cooperation skills, towards smarter systems that will learn and interact with their environment, collaborate with humans, plan their future actions, and execute tasks with immense precision (Salzman and Stern 2020). In this way, today's society is frequently confronted with new "groundbreaking" AI models based on advances in algorithms and computing. These advances are directly owing to the growing consumption of data, computing, and energy resources.

With the rising consumption of data and resources, it is important to emphasize that growth can also be achieved through efficient utilization of data and a deeper understanding of current AI paradigms. *This means, understanding the optimization of parameterized, large-scale distributed AI models, i.e., understand the training of AI models. Also, how to make the most efficient use of computational resources, identifying sources of errors and limitations in highly parallelized training methods, etc. It is consequently imperative that we understand how model training by asynchronous and multi-agent methods is affected by training errors and how to compensate for them.* Broadly speaking, these are the motivations and topics of the presented thesis. One important type of training error considered herein is due to information delays that arise from poor communication and asynchronous computation. Such errors affect the performance of parallelized training algorithms in single and multi-agent learning. The information delays arise from asynchronous computing, communication, or other information-sharing effects. The core

contribution herein is an in-depth study of information delays and an analysis of a broad class of discrete-time distributed algorithms under the coupled stochastic approximation (SA) framework.

The theoretical results to be developed in this framework are based on powerful tools from applied mathematics. The first core tool is a dynamical systems perspective on discrete-time iterative SA algorithms (Michel Benaïm 1996). Discrete-time algorithms are studied by relating them to their continuous-time counterparts. In particular, discrete-time algorithms can be shown to closely track solutions of continuous-time dynamical systems. Then, dynamical systems theories, e.g., viability theory (Aubin and Cellina 2012) – the study of dynamical systems with viable constraints, are used to draw conclusions about the dynamical system's behavior, thence the actual algorithm. The standard form of an algorithm x_n operating in $\mathbb{R}^d$ studied herein is given by the recursion

$$x_{n+1} = x_n + a(n)\left[h(x_n) + e_n + M_{n+1}\right], \tag{1.1}$$

with a positive stepsize $a(n)$, mean algorithm drift $h(\cdot)$, information delay error process e_n and sampling noise process M_{n+1}. Based on this algorithm form, the second core tool employed is modeling noise, errors, and especially information delays as stochastic processes, i.e., sequences of random variables (Durrett 2019). Various tools from applied probability will then be used to characterize sufficient conditions for SA algorithm analysis. In particular, martingale convergence theory is used to study sampling noise (Pollard 2002), and strong mixing theory is used to describe the temporal dependence of events that give rise to information delays (Bradley 2005).

The prototypical distributed algorithm that yields the standard form (1.1) is a D agent ($D \in \mathbb{N}$) discrete-time system with D coupled, possibly asynchronous stochastic iterations given by:

$$x^i_{n+1} = x^i_n + a(n) g^i\left(x^1_{n-\tau_{i1}(n)}, \ldots, x^i_{n-\tau_{ii}(n)}, \ldots x^D_{n-\tau_{iD}(n)}; \xi^i_{n-\Delta^i(n)}\right), \quad n \geq 0,\ i \in D, \tag{1.2}$$

with x^i_0 an initial vector that is often randomly chosen, and where

- $x_n \coloneqq (x^1_n, \ldots x^D_n)$ in $\mathbb{R}^d \coloneqq \mathbb{R}^{d_1} \times \ldots \times \mathbb{R}^{d_D}$, $n \geq 0$.
- $x^i_n \in \mathbb{R}^{d_i}$ is the real-dimensional value of the iteration on system i at discrete time step n.
- $\{\xi^i_n\}$ is a sequence of random observations used to update the iteration on system i.
- $g^i(\cdot)$ is the local drift function of iteration i that describes how iteration i changes from one time step to the next one.
- $\{a(n)\}$ is a sequence of positive numbers, referred to as the stepsize sequence, i.e., $a(n)$ weights the local drift functions to update the iterations at step n.
- $\tau_{ij}(n)$ denotes the random information delay called *Age-of-Information* (AoI) that occurs as system i uses the value of system j from the old time step $n - \tau_{ij}(n)$ to evaluate its local drift at time n.
- $\Delta^i(n)$ denotes the random information delay called *Data Age-of-Information* (DataAoI) that occurs as systen i uses the sample from its observation process from the old time step $n - \Delta^i(n)$ to evaluate its local drift at time n.

Distributed iterations, such as (1.2), are fundamental to multi-agent reinforcement learning (MARL) (Lowe et al. 2017), distributed asynchronous computing (Zhou et al. 2022), distributed control and estimation (P. Bianchi, Fort, and Hachem 2013), and distributed optimization algorithms (Ramaswamy, Redder, and Daniel E Quevedo 2021a).

Example 1.0.1 (Distributed Asynchronous Stochastic Gradient Descent (DASGD))**.** *It is an important example of* (1.2)*. The goal of DASGD is to find a local minimum of the stochastic optimization problem* $\min_{x\in\mathbb{R}^d} F(x)$ *with objective* $F(x) \coloneqq \int_\Xi f(x;\xi)d\mathbb{P}(\xi)$ *for some random function* $f : \mathbb{R}^d \times \Xi \to \mathbb{R}$*. Here,* $f(x;\xi)$ *is the observed objective function at a sample* $\xi \in \Xi$*. In machine learning applications, the sample space* Ξ *represents a dataset. If we take the example of LLM applications,* $f(x;\xi)$ *will be a loss function,* x *will be a neural network (NN) parameter vector, and* ξ *will be the datapoint being processed. When training such models using the distributed computing paradigm, gradients have to be calculated for subspaces* $\mathbb{R}^{d_1} \times \ldots \times \mathbb{R}^{d_D}$ *of the whole optimization space to ensure feasible training. Multiple asynchronous machines will work on each subspace to accelerate the training (B. Yang et al. 2021). The local objective (drift) function in* (1.2) *is then simply the sample component gradient:* $g^i(x;\xi) \coloneqq \nabla_{x^i} f(x;\xi)$*. Moreover, the mean drift in* (1.1) *is then given by the expected concatenated sample gradients* $h(x) \coloneqq \mathbb{E}_\xi\left[\left(g^1(x;\xi),\ldots,g^D(x;\xi)\right)\right]$*. Further, the AoI random variables* $\tau_{ij}(n)$ *naturally arise since each component may be updated multiple times when a single machine computes a sample component gradient. Finally, DataAoI can occur if the training data is communicated to the learning system from sensors or is only made available after some time.*

In this thesis, we[1] study coupled iterations such as (1.2) that describe how individual agents/ subsystems update local parameters based on information about the other iterations and locally available data. The information may arrive with a potentially large delay – *Age of Information* (AoI). This AoI can occur due to various phenomena. Two sources of AoI are asynchronous parameter updates on distributed computing infrastructure as in Example 1.0.1 and communication between physically distributed systems. DataAoI occurs in distributed, communication-limited/resource-constrained scenarios (e.g., sensor networks), where obtaining up-to-date information is costly, thus systems have to use/reuse outdated data for local updates. We analyze the impact of errors due to such delays within the context of a general class of iterative methods called stochastic approximation algorithms (M. Benaïm 2006). This class originates from seminal works on root-finding iterations and dynamic programming from Robbins and Monro (1951), Kiefer and Wolfowitz (1952), and Bellman (1957). For the last seven decades, SA has been of

[1] Whether or not to use the pronoun "we" in a Ph.D. thesis is a common topic of debate. In my opinion, it's a stylistic choice, and I prefer to write and read English in a more active voice. I will therefore use "we" because I don't like the use of "I" and because I like the idea of an engaging conversation between the writer and the reading audience. I apologize if the frequent use of "we" irritates you (the reader); you may substitute "I" for "we", which would be technically correct, albeit "we" acknowledges the valuable guidance of my advisors throughout my work.

renewed interest due to various applications in signal processing, economics, game theory, machine learning, and optimization (Harold, Kushner, and G. Yin 1997; Benveniste, Métivier, and Priouret 2012; Bravo 2016; Lei et al. 2020; S.. Bhatnagar, Prasad, and Prashanth 2013; Uryasev and Pardalos 2013; Ghadimi and Lan 2012).

For SA iterations, three questions are of fundamental interest (V. Borkar 2022): 1) Is the SA algorithm numerically stable, i.e., is $\sup_{n\geq 0}\|x_n\| < \infty$ almost surely, or, equivalently, is every iteration bounded by a sample path-dependent compact set? 2) Does the iteration converge, and are the limit points elements of an invariant set of the drift function $g(\cdot)$? 3) At what rate does the iteration converge? A non-zero rate of convergence estimate implies that the iteration converges, which in turn implies that the iteration is almost surely stable. Hence, one will typically establish an iteration's stability, followed by a convergence and a rate of convergence analysis. *Broadly speaking, this thesis studies distributed SA algorithms and the AoI processes affecting them. It presents verifiable sufficient conditions that ensure iterations are stable and convergent.*

To answer such questions for distributed iterations, we study properties and conditions for the random AoI and random DataAoI sequences $\tau_{ij}(n)$ and $\Delta^i(n)$ in (1.2). We will collectively refer to these sequences as AoI processes and will soon make precise what we mean by an AoI process. Fundamentally, we ask: How "large" can AoI be while still ensuring stability and convergence of distributed iterations? How does AoI affect the choice of stepsize sequences (learning rates) and the convergence rate? Even more fundamentally, we ask: What is AoI? Specifically, how can we define AoI as a stochastic process? What verifiable assumptions can be imposed on AoI processes, and what properties useful for SA analysis can be derived?

We answer the aforementioned questions over the course of this thesis. In Section 1.4, we will assign substantiated versions of these questions to chapters that address their respective answers in the thesis. Finally, the structure overview Figure 1.7 on page 27 will highlight the topics and connections between the chapters. We will now further motivate the thesis and present the concrete iterations and problems to be covered.

1.1 Motivation

Distributed SA provides a generic, yet effective, mathematical framework to study both asynchronous optimization and distributed multi-agent learning. Such a study is imperative to accelerate distributed learning via distributed optimization that uses asynchronous heterogenous computing resources. The framework can be applied to study empirical risk minimization (ERM) problems involving deep neural networks (Montanari and Saeed 2022) - the most common statistical learning paradigm in machine learning. It also applies to sequential data-driven

decision-making, control, and reinforcement-learning (RL) problems in stochastic environments (S. Meyn 2022). In the case of ERM, a parameterized model - deep neural network - is trained to minimize an empirical objective function using a given data set. By contrast, in the sequential decision making case, the data used to train the model is usually generated online by the model itself. This is done by trying out model decisions for various scenarios over time. For both applications, asynchronous implementations and parallel computing resources can significantly accelerate learning.

1.1.1 Acceleration with asynchronous computing

Accelerating machine-learning algorithms via distributed computing (DC) has become a critical technique to quickly train huge AI models such as large language models, using massive amounts of data (Huang et al. 2019; Narayanan et al. 2019; Chowdhery et al. 2022). Acceleration is achieved by scheduling multiple workers (computing nodes/processors) to update a machine-learning model asynchronously (Dean et al. 2012; Ben-Nun and Hoefler 2019). For example, in parameter-server systems, workers sequentially read the parameters of a model and a minibatch from a data set to then asynchronously compute optimization steps (e.g., stochastic gradients) to update the model. Asynchronicity here refers to workers that compute and apply updates independently without waiting for other workers; e.g., worker 1 may have computed updates on five minibatches while worker 2 is still working on its first minibatch. In other words, updates are applied without any bounds on the order of operation. Such lightweight methods with small coordination overhead and asynchronous operation have been shown to reduce memory usage while drastically increasing processor utilization to train deep learning and large language transformer models (S. Zhang et al. 2013; Guan et al. 2019; B. Yang et al. 2021). Lightweight methods will become even more useful in envisioned volunteer computing networks where users only offer a small fraction of their computing resources to other users, resulting in highly heterogeneous computing infrastructure (Anderson 2020).

Accelerated learning through lightweight asynchronous computing methods comes at the cost of parameter update errors that arise when updates are computed based on outdated information (Zhou et al. 2022). The prototypical DC scenario that yields this interaction is illustrated in Figure 1.1 for an ASGD implementation (described in Example 1.0.1), where multiple machines update a single component. It can be expressed in the general form (1.2) using a single AoI sequence $\tau(n)$. The ASGD iteration then reads

$$x_{n+1} = x_n - a(n)\nabla_x f(x_{n-\tau(n)}; \xi_n), \tag{1.3}$$

which corresponds to the "master iteration" in Figure 1.1. The AoI $\tau(n)$ from the perspective of the master iteration (1.3) arises when multiple systems compute updates that are applied

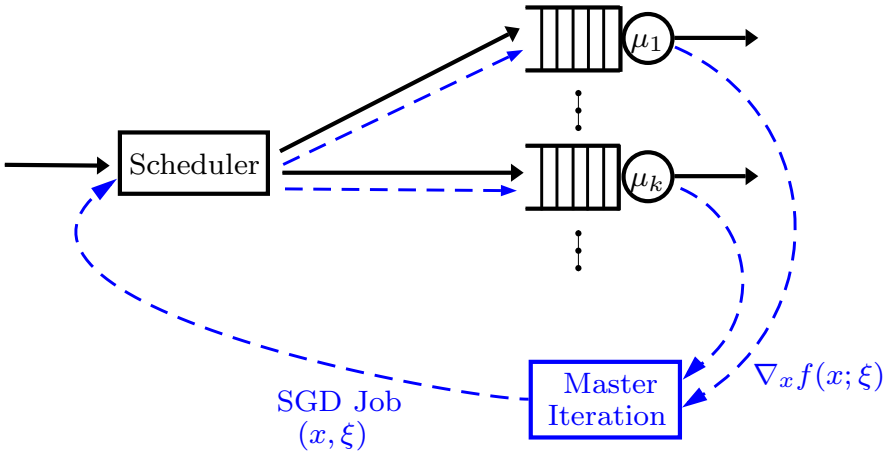

Figure 1.1: DC system running ASGD concurrently with other jobs. Workers are represented as queues with service rate μ_k, local scheduling policies, e.g., first-come-first-serve, and potential priority policies. The queues sequentially receive SGD jobs from an SGD master iteration (left blue arrows). The jobs are queued, and once completed, the SGD update steps are returned to the master iteration (right blue arrows). In addition, other jobs are assigned to the queues by the scheduler.

without synchronization. More generally, distributed asynchronous iterations such as DASGD can be written in the form (1.2) affected by AoI, which will be discussed in-depth in Chapter 2. It has been noticed that the presence of AoI reduces the performance of the methods when inappropriate hyperparameters - typically stepsizes - are chosen (Lian et al. 2015). To avoid this problem, precise information about AoI is essential to optimize and predict the performance of asynchronous iterative methods. This makes it pertinent to characterize how processing times and asynchronous computing on DC systems give rise to AoI to design effective methods that maximize resource utilization while guaranteeing performance.

The stochastic nature of the processing times of asynchronous parameter updates in parallel DC infrastructure is due to real-world system aspects like queuing, priorities, preemption, heavy-tailed traffic, and advanced workload managers (Georgiou 2010; Clavier et al. 2020; Tirmazi et al. 2020). Notably, these aspects can result in unbounded heavy-tailed stochastic processing times. Specifically, the random variables associated with the processing times may have *unbounded* first moments. For example, this has been observed in the case of asynchronous algorithms run on public clouds (Samsi et al. 2021). This happens because the ASGD machine-learning method is run as a low-priority sequential job concurrently with other jobs on a cloud server that allocates a potentially time-varying set of workers to the method, as illustrated with the scheduler in Figure 1.1.

Various questions are of prime interest for asynchronous parallel computing scenarios like ASGD. Foremost, *how should we choose the stepsize of asynchronous methods executed on parallel computing infrastructure?* The typical information to answer such questions includes the number

of available computing resources, the deployed job scheduling policies, and the processing time data of similar sequential computing jobs as well as data from typical concurrent jobs run on a cluster. Such processing time distribution information may be historical or from recently fished jobs on the computing infrastructure. In addition, advanced information like the dependency of jobs sequentially scheduled to workers, the time-dependent workload of a cluster, etc., can be tracked with modern workload managers like slurm (Georgiou 2010). With such sets of information, predicting AoI properties to be expected for ASGD training prior to runtime is challenging. Fundamentally, the discrete-time, discrete-valued AoI sequences affecting an asynchronous algorithm must be derived in closed form from the continuous-time processing times of jobs. In addition, it is unclear how to describe and account for dependencies in the processing time of jobs, which often occur because users submit similar jobs. Prior to this work, there were no tools to attempt such challenges. To resolve this gap in the literature, we will present, on the one hand, new results for the stability and convergence of distributed SA affected by AoI. On the other hand, we present fundamental models and characteristics for AoI processes that yield verifiable conditions to apply the developed stability and convergence theory to distributed asynchronous computing methods based on processing time and infrastructure information.

1.1.2 Distributed multi-agent learning and optimization

Besides asynchronous computing, a motivation for the present work is learning and optimization of physically distributed multi-agent systems (MAS). These MAS occur, for example, in robotics (Salzman and Stern 2020) and edge-computing networks (Sofla et al. 2022). For such applications, AoI can arise due to asynchronous computing resources and delayed communication between physically distributed systems. Delay in communication is often due to resource limitations, for instance, energy constraints in remote battery-powered wireless sensor networks (He et al. 2020). In the most extreme scenario, systems must withhold the exchange of information for as long as possible to minimize their energy consumption or to satisfy privacy restrictions as in federated learning (C. Zhang et al. 2021). The natural question is, *what is the least frequency with which information sharing is permitted so that distributed systems can still effectively and optimally solve multi-agent learning problems in a decentralized manner?*

In multi-agent learning algorithms, agents often exchange parameterizations of local models. Since communication is required in physically distributed systems, the algorithms must account for AoI. Additionally, AoI may arise as the agents receive delayed observations from faraway sensors or even from other agents. In Equation (1.2), we referred to this AoI as DataAoI since it captures the *age of the observations*. For DASGD, described in Example 1.0.1, DataAoI is usually not a concern as data is typically assumed to be independent and identically distributed (i.i.d.). However, for MARL, data is Markovian; hence, DataAoI has an impact. When the observations

depend on the decision of a multi-agent learning algorithm, it is unclear how much DataAoI can be tolerated such that the algorithm still converges. In other words, when the learning agents gather experience from old samples along their training trajectory, does the accumulated experience represent "well" how the agents currently act in the environment? If not, the learned policies could be biased toward an agent's old behavior, which is usually undesirable.

We will now introduce details on the core setting considered in this work to state the problems and the acquired solutions. Further, we begin the discussion of AoI as a stochastic process.

1.2 Stochastic approximation algorithms

Traditional SA is centralized in nature, implemented as a single iteration to, e.g., find the roots of a map $h : \mathbb{R}^d \to \mathbb{R}^d$. The iteration starts from an initialization $x_0 \in \mathbb{R}$ and then follows an incremental update rule

$$x_{n+1} = x_n + a(n)\big[h(x_n) + M_{n+1}\big], \quad \text{for } n \geq 0, \tag{1.4}$$

where $\{a(n)\}$ is a positive stepsize sequence, and M_n is a martingale difference sequence (Definition 0.0.9) due to the use of samples. Further, the drift h usually represents the mean drift of an underlying sampling-based iteration, e.g., (1.2) without AoI and i.i.d. samples is of the form (1.4) with $h(x) \coloneqq \mathbb{E}_\xi\left[g(x;\xi)\right]$.

Depending on how h is defined, iteration (1.4) becomes a fixed point finding method - pertinent to dynamic programming and reinforcement learning (D. P. Bertsekas and J. N. Tsitsiklis 1996). The drift h can also be defined such that various gradient-based optimization algorithms (S.. Bhatnagar, Prasad, and Prashanth 2013) are described using (1.4). Hence, (1.4) is a very generic iteration that serves as a powerful analytic framework. The theory of SA was placed in a rigorous mathematical dynamical systems framework in the seminal work of Michel Benaïm (1996), which in turn is based on the breakthrough work of Ljung (1977). Ljung first established that SA algorithms could be asymptotically associated with solutions to deterministic ordinary differential equations (ODEs) under reasonable assumptions. This constitutes the *dynamical systems perspective*, via the *ODE method*, of stochastic approximation algorithms. In this work, we take the dynamical systems perspective to study component-wise distributed versions of (1.4). The notation used throughout follows the modern treatment of SA from a dynamical systems perspective by V. Borkar (2022).

The main idea of the dynamical systems perspective is to study properties such as stability and convergence of (1.4) via solutions of the associated ODE $\dot{x}(t) = h(x(t))$. The standard procedure is first to establish the stability of (1.4) followed by a convergence characterization. There are various schemes to establish the stability of SA algorithms; see M. Benaïm (2006),

Harold, Kushner, and G. Yin (1997), V. Borkar (2022, Chapter 4) and the reference therein. One of the most attractive schemes is the stability through scaling approach proposed by V. S. Borkar and S. P. Meyn (2000), which is now known as the Borkar-Meyn Theorem (BMT). This scheme is attractive as its assumptions can be verified solely using the algorithm drift h.

The BMT establishes the stability of (1.4) by studying a family of ODEs with scaled dynamics $h_c(x) := \frac{h(cx)}{c}$ for $c \in [1, \infty]$ under the following condition, which implies that the limiting dynamics of $h_c(x)$ as $c \to \infty$ is eventually attracting towards the origin.

Condition 1 (*BMT stability condition*)**.** *The functions $h_c(x) \to h_\infty(x)$ converge uniformly on compact sets as $c \to \infty$ for some $h_\infty \in \mathcal{C}(\mathbb{R}^d)$. Furthermore, the ODE $\dot{x}(t) = h_\infty(x(t))$ has the origin as its globally asymptotically stable equilibrium (Definition 0.0.5).*

V. S. Borkar and S. P. Meyn (2000) established that under the BMT Condition 1 and additional standard assumptions (to be discussed later on), a rescaled, interpolated version of the iteration x_n will be asymptotically close to solutions of the limiting ODE $\dot{x}(t) = h_\infty(x(t))$. Then, if x_n were unstable, x_n would potentially have to make arbitrarily large jumps in finite time, as $\dot{x}(t) = h_\infty(x(t))$ is asymptotically stable. This contradicts the classical discrete version of Gronwall's inequality, and the BMT follows. In Chapter 2, we present a more detailed description of the BMT proof.

Based on the idea of the BMT, several generalizations have been proposed. Shalabh Bhatnagar (2011) proposed a generalization to asynchronous SA, Lakshminarayanan and Shalabh Bhatnagar (2017) generalized the idea to two-timescale SA iterations, and finally Ramaswamy and Shalabh Bhatnagar (2017) presented an important generalization to SA with set-valued dynamics. The generalization of Shalabh Bhatnagar (2011) was proposed for bounded delays, which poses a significant restriction. We will see in Chapter 7 that for a natural asynchronous computing model, the resulting AoI will generally have unbounded support due to stochastic processing/service times. *In other words, the hitherto presented literature lacks a distributed version of the BMT for unbounded information delays representative of asynchronous computing and distributed learning settings.*

1.2.1 Distributed asynchronous stochastic approximation algorithms

Centralized implementations, such as standard SA iteration (1.4), can suffer from computational bottlenecks or can be infeasible due to the decentralized nature of a problem. Therefore, distributed asynchronous stochastic approximation (SA) algorithms were developed, where multiple systems/nodes/agents interact to find a function's roots (D. Bertsekas and J. Tsitsiklis 2015). Such distributed asynchronous parallel implementations of SA algorithms were first considered for stochastic gradient-based methods (J. Tsitsiklis, D. Bertsekas, and Athans 1986). Distributed

SA refers to iterations executed via computer networks, which are thus affected by communication delays. Asynchronous SA traditionally refers to iterations that run with different clocks, such that at every time step, only some iterations are updated (V. S. Borkar 1998). *More recently, asynchronous SA refers to algorithms where many computing systems update one or many components of a single global set of variables. This is the definition relevant to this thesis.* In such scenarios, AoI arises as discussed Section 1.1 and exemplified with (1.3), even for a single global variable updated by many systems. We conclude that the dominant feature in these asynchronous computing, distributed learning, and distributed optimization algorithms is that a set of variables is updated as a function of old versions of itself, which we described by AoI random variables. The natural distributed SA version of (1.4) that can represent many of these algorithms can be stated as follows.

Starting from some $x_0 \coloneqq (x_0^1, \dots x_0^D)$ in $\mathbb{R}^d \coloneqq \mathbb{R}^{d_1} \times \dots \times \mathbb{R}^{d_D}$, each component i is updated with the recursion

$$x_{n+1}^i = x_n^i + a(n)\big[h^i(x^1_{n-\tau_{i1}(n)}, \dots, x^D_{n-\tau_{iD}(n)}) + M_{n+1}^i\big], \quad \text{for } n \geq 1, \tag{1.5}$$

where $h^i : \mathbb{R}^d \to \mathbb{R}^{d_i}$ are local drift functions; $\tau_{ij}(n)$ denote AoI random variable as in (1.2); M_n^i are local Martingale noise sequences; $\{a(n)\}$ is the positive stepsize sequence.

When all x_n^i are almost surely stable and h is Lipschitz continuous (Definition 0.0.1), then showing the convergence of (1.5) can be done by a simple reduction of the iteration (1.5) to a version of the standard SA iteration (1.4). Specifically, (1.5) can be written as

$$x_{n+1} = x_n + a(n)\big[h(x_n) + e_n + M_{n+1}\big] \tag{1.6}$$

with an error term

$$e_n \in \mathcal{O}\left(\sum_{i,j} \sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)\right) \text{ a.s.} \tag{1.7}$$

Then, after recalling the notion of stochastic dominance from Definition 0.0.6, consider the following AoI condition:

Condition 2. *(Previous AoI condition) All $\tau_{ij}(n)$ are stochastically dominated by a random variable $\overline{\tau}$, denoted by $\tau_{ij}(n) \leq_{st} \overline{\tau}$ for all $n \geq 0$, with $\mathbb{E}\left[\overline{\tau}^p\right]$ for some $p > 1$.*

Under Condition 2, if $a(n) \in \mathcal{O}\left(n^{-\frac{1}{q}}\right)$ for some $q \in [1, 2)$, with $q \leq p$, it follows from the Borel-Cantelli Lemma that $e_n \in o(1)$. The convergence of (1.6) then follows along the lines of the convergence proof for (1.4) under standard assumptions using that e_n vanishes asymptotically (V. Borkar 2022, Section 2.2 & 6.3).

The strongest assumption for the outlined convergence is that x_n is stable. Without stability, (1.7) holds a priori only if the drift function $h(x)$ is bounded, a serious restriction. In addition,

even for bounded drift, the stability result of Shalabh Bhatnagar (2011) only holds for bounded AoI. Still, even when stability is assumed or guaranteed, the bounded p-th moment requirement in Condition 2 is another restriction, as illustrated by the following example.

Example 1.2.1. *Consider a synchronized distributed learning scenario where D agents execute iteration (1.5). Suppose the agents exchange their iteration values sequentially in a peer-to-peer manner (or in accordance to a strongly connected directed communication graph with an appropriate forwarding mechanism). The number of steps required for information exchange between a pair of agents can be well represented by a lattice renewal process (Definition 0.0.10) (or a more general point process on $\mathbb{R}$). Then, the resulting AoI will be given by the corresponding backward recurrence times of the renewal processes (Definition 0.0.12). Finally, one can show (see, e.g., Serfozo (2009)) that the resulting AoI will satisfy Condition 2 for $p > 1$ if and only if the renewal process interarrival times have bounded $(p+1)$-th moment.*

We conclude from Example 1.2.1 that heavy-tailed interarrival time distributions with infinite variances, which in turn implies infinite first moments for the AoI process, are not analyzed by the available SA literature with respect to both stability and convergence. Such scenarios occur in parallel computing due to job resource requirements that are heavy-tailed (Tirmazi et al. 2020; Samsi et al. 2021), in communication due to heavy-tailed interference as, e.g., in large internet-of-things systems (Clavier et al. 2020), and in general due to the nature of bursts occurring in systems affected by human dynamics (Barabasi 2005).

Based on the discussion so far, the following are some of the key research questions answered in this thesis:

(Q1) What version of the Borkar-Meyn stability theorem holds for distributed SA algorithms with *AoI's that have infinite first moments*?

(Q2) To answer (Q1), we must first answer: is $p > 1$ in Condition 2 necessary for stability and convergence, or can we obtain a similar condition for the $p \in (0, 1]$ case? What will be the trade-off here? How are the various algorithm parameters affected, e.g., how should we restrict the learning rate? and how is the convergence rate affected?

(Q3) What properties must DataAoI possess, so that, in distributed multi-agent learning, the learned policies are not biased towards the agent's old behavior?

(Q4) How can we ensure the existence of stochastically dominating random variables? Can we do better and characterize the exact AoI distribution in different asynchronous computing and distributed learning settings? How to guarantee Condition 2 for some desired $p \in (0, \infty)$?

To answer these questions, especially (Q4), we need to first view AoI as an appropriate stochastic process.

1.3 Age of Information (AoI)

In the previous section, we discussed that AoI causes drift errors, potentially jeopardising stability and convergence. Further, we saw in Example 1.2.1 that AoI may arise due to subsequent communication which can be well represented by renewal processes. In general, there are many sources of AoI. We are interested in the underlying fundamental structure and properties that unify many of these sources. In the end, we want to derive conditions a) that are practically verifiable in a real setting, and b) that suffice for stability and convergence of distributed SA iterations.

Historically, AoI has been popularly used in point process theory to study the age distribution of populations as individuals/systems live and die (Feller 1941; Doob 1948; Cox and Isham 1980). In information theory and computer networking, AoI has recently become popular as a representative metric for the freshness of data, where AoI has been studied for various queuing and scheduling models (Yin Sun et al. 2019). For a point process, AoI describes the time since the last point occurred. In other words, the AoI is simply the backward recurrence time (Definition 0.0.12) of the point process modeling the communication.

If we consider AoI within the context of wireless communication, one might expect it to be "small" and bounded, especially in the current 5G/6G era (Viswanathan and Mogensen 2020). However, the communication delay can still add up for high-dimensional data (e.g., when transmitting neural network parameters). Further, the communication network may only provide limited access due to interference in dense environments (Clavier et al. 2020), and the successive wireless network access can be highly correlated (Boban, Gong, and W. Xu 2016). In summary, we observe that a network model that captures dependent communication and network access is necessary to describe AoI processes in representative real-world scenarios. Instead, most studies of AoI in networking focus on i.i.d. interarrival times (Yates et al. 2021), and the only available studies of point process with dependent interarrival times do not enable a way to quantify dependency (Kombrink 2018).

Communication is just one possible source for AoI. Generally, information access may be well-represented by a sequence of events representing whether or not new information arrives. A crucial component should be that the sequences of events may be dependent. These observations about the presence of dependency in communication, which is not well covered in the literature, motivated the study of AoI processes as a function of event processes with dependency decay.

1.3.1 AoI processes driven by event processes with dependency decay

As many effects give rise to AoI, we seek to describe them with a fundamental model for the information exchange between two nodes, e.g., a source and a monitor. We describe information

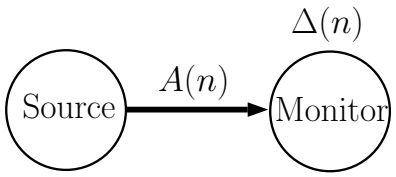

Figure 1.2: A source sends status updates through a channel to a monitor. At time $n+1$ an update is successfully received, and the AoI process $\tau(n)$ is reset to one if the event $A(n)$ has occurred.

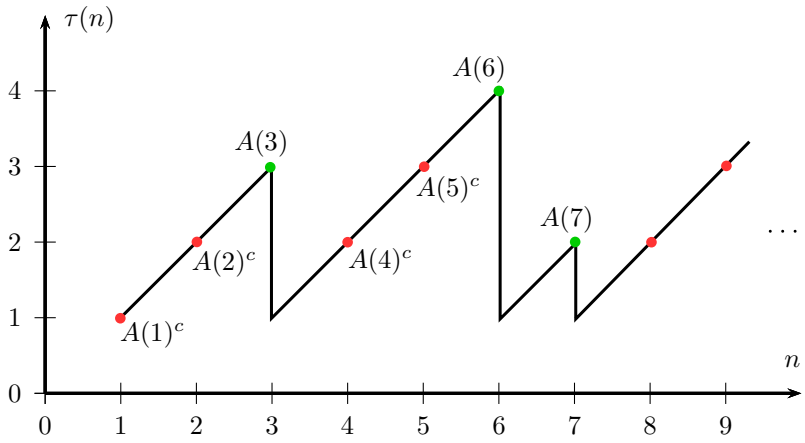

Figure 1.3: Illustration of a sample path for the fundamental AoI process (1.8). Green dots and red dots mark the occurrence and non-occurence of an event in the preceding interval, respectively.

exchange by a sequence of events $A(n)$ representing successful status updates from the source received at the monitor (Figure 1.2). We refer to a *slot* n as the interval from discrete time step n to $n+1$. An update sent at time step n is received either at time step $n+1$ (more precisely, at the start of time slot $n+1$) or not at all. At first, this model is a limitation to general AoI sequences, but we will see in a second how it can be used to describe more general AoI.

Based on the event sequence $A(n)$, we will now describe a fundamental AoI process that we denote by $\tau(n)$. Intuitively, $\tau(n)$ grows at a unit rate and resets to one at the occurrence of events. If an update sent at time step n is received at time step $n+1$, then we say the event $A(n)$ has occurred. The event $A(n)$ is thus associated with the n-th time slot. Hence, whenever an event $A(n)$ occurs, $\tau(n+1) = 1$. The resulting fundamental AoI process $\tau(n)$ is

$$\tau(n+1) \coloneqq \begin{cases} 1, & A(n) \text{ has occurred,} \\ \tau(n)+1, & \text{otherwise,} \end{cases} \tag{1.8}$$

with $\tau(0) \coloneqq 0$. This simple AoI model thus considers status updates that require a single time slot for communication. The AoI process (1.8) can be seen as a special random walk on the positive real line that restarts whenever an event occurs. From this perspective, the AoI process can represent various hill-climbing phenomena where a process grows over time and then resets at certain events. This has, for example, been used in physics to study Doppler laser cooling,

where atoms rise from a ground level in the presence of a light field (Montero and Villarroel 2016). We, therefore, expect general interest in the study of 1.8 beyond information delays.

In previous studies, such as (Montero and Villarroel 2016), event processes are assumed as i.i.d. Bernoulli process. Instead, we will consider $\tau(n)$ when $A(n)$ merely admits dependency decay over time. Loosely speaking, this means that the occurrence of events $A(n)$ and $A(m)$ becomes less dependent as $|n-m| \to \infty$. In wireless communication, $A(n)$ may represent the joint event that information is sent and that the used communication channel is in a good state.

For the AoI model (1.8), a pertinent question is which AoI processes can be represented by it and whether all AoI processes have a sawtooth-like sample path as in Figure 1.3. For example, consider point processes without multiple occurrences (such as renewal processes in Example 1.2.1), represented by a sequence of interarrival times $W(n)$, $n \geq 1$, such that $W(n)$ is the number of timeslots between the $n-1$-th and the n-th status update.

Example 1.3.1. *Consider a lattice point process, such that $W(n) \in \mathbb{N}$. An example sample path is shown in Figure 1.4. The resulting AoI process can be stated using the event sequence*

$$A(n) \coloneqq \Big\{ \sum_{i \leq k} W(k) = n \text{ for some } k \Big\}. \tag{1.9}$$

The resulting AoI process is

$$\tau_r(n) \coloneqq \begin{cases} W(k(n)), & A(n), \\ \tau_r(n-1)+1, & A(n)^c, \end{cases} \tag{1.10}$$

with $k(n) \coloneqq \sup\{k \in \mathbb{N} : \sum_{i \leq k} W(i) \leq n\}$. Observe that whenever $A(n) = 1$, the AoI process is set to $W(k(n))$, the time required for the information exchange.

From Figure 1.3 and Figure 1.4, we can see that the fundamental AoI process (1.8) cannot represent (1.10) as the backward recurrence time of the point process does not reset to 1, but to the time required for the information exchange. However, it turns out that

$$\tau_r(n) = \tau(n - \tau(n)) + \tau(n) - 1, \tag{1.11}$$

where $\tau(n)$ is the fundamental AoI process defined using (1.9). In other words, the fundamental AoI process can be used to describe the AoI arising from point processes represented by a sequence of interarrival times. In this way, the fundamental AoI process serves as a basis for describing more complex AoI processes, e.g., the AoI arising from asynchronous computing. More importantly, the analysis of Equation (1.8) presented herein will consider event sequences $A(n)$ that merely admit dependency decay over time described by strong mixing notion. With this, the fundamental AoI process can be used to describe AoI due to interference in wireless communication and AoI in asynchronous computing with correlated jobs.

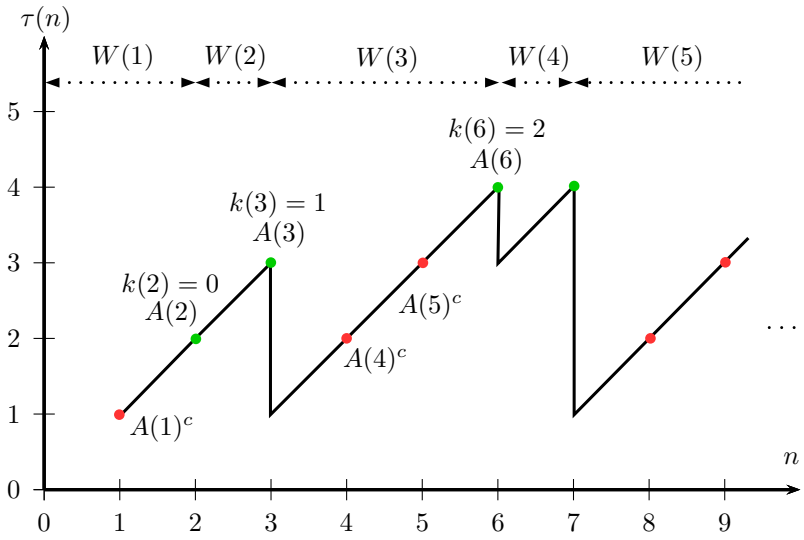

Figure 1.4: Illustration of a sample path for an AoI process described by interarrival times.

1.3.2 How AoI arises from asynchronous computing

For AoI caused by asynchronous parameter server updates, the most common examples are ASGD and its variants (Netrapalli 2019). These methods fall into a general class of asynchronous distributed parameter server implementations (see Algorithm 2 and Algorithm 3 in Section 7.2.1). Asynchronous methods typically start by initializing parameters, e.g., a machine learning model. Workers then read the model parameters and a sample from a data set to compute an update step for the parameters (typically for a subspace of the parameter space (Raina, Madhavan, and Ng 2009; Guan et al. 2019)). The update steps are then returned to the parameter server, which applies the parameter update. While a worker computes and sends an update step, other workers may have updated the parameters several times.

In Section 1.1.1, we explained how such asynchronous implementations lead to accelerated optimization at the cost of received updates that have already aged when applied. In fact, the difference between the current parameter iteration index and the index to which the computed update step corresponds is precisely the current AoI.[2] For example, suppose a worker uses the parameter x_m from step m to calculate an update step, but this update arrives at the server at iteration index n (for some $n \geq m$) and is used to update x_n. The resulting AoI of this update is thus $\tau(n) = n - m$. Let $m(n)$ be the index from which an update step applied at index n was computed. Then, $\{\tau(n)\}$ (with $\tau(n) \coloneqq n - m(n)$) is the resulting sequence of AoI random

[2] Alternatively, one can describe separate AoI sequences for each system computing updates for a parameter iteration; the separate AoI sequences simply count the number of updates made, while one system computes an update. Then, when one system applies an update, the AoI associated with the parameter update is just the corresponding separate AoI of the system. This decomposition into AoI sequences associated with individual systems is useful for analysis; see Chapter 7.

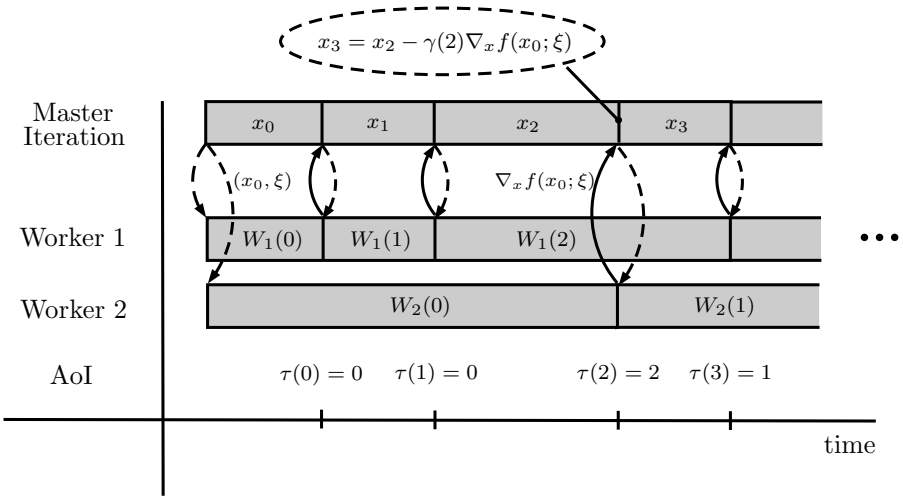

Figure 1.5: Illustration of AoI arising for ASGD with two workers. $W_k(n)$ is the n-th time to compute an update step by worker k. Dashed lines: exchange of SGD jobs. Solid lines: exchange of stochastic gradients. $f(x;\xi)$ denotes the loss function for a data sample ξ as in Example 1.0.1.

variables for an asynchronous parameter server iteration. An illustration for an AoI sequence generated by ASGD with two workers is presented in Figure 1.5.

The introduced definition of AoI in asynchronous computing can now be naturally extended to asynchronous computing with component-wise parameter updates on subspaces. This leads to the definition of AoI sequences $\tau_{ij}(n)$ as used in 1.2. A natural model for the updates of processors on each component is to use point processes or renewal processes represented by a sequence of processing times as in Figure 1.5. A crucial factor for this model is that events are not aggregated compared to the study of parallel renewal processes in neural biology (Cox and Smith 1954; Shanechi et al. 2012), which are therefore not applicable. What is important here is the number of events (parameter updates) that occur from other renewal processes while one renewal process is waiting for its next event to happen. Because of this property, one can suspect that the AoI caused by asynchronous computing is not dependent on the backward recurrence times of the point processes but merely on the length of each computing interval. To verify this, one has to analyze the distributional limit of the AoI caused by parallel point processes, specifically, the discrete number of updates by other workers in continuous time while one worker computes an update.

In summary, we saw in this section that event processes (1.8) appear to be representative models for AoI processes but have not been studied for dependent communication. Further, we explained how AoI arises in asynchronous computing due to updates made by other workers while one worker is computing an update step, which parallel point processes may well represent. From the discussion so far on AoI described by event sequences and asynchronous computing, the following questions arise:

(Q5) What fundamental structure (e.g., saw-tooth-like sample paths) do AoI sequences possess that can be used to define AoI processes in general?

(Q6) What growth properties do AoI sequences have as a function of given moment bounds?

(Q7) How can we describe the dependency in event sequences and interarrival time sequences of point processes? Further, what properties of AoI processes can be concluded from a dependency model?

(Q8) What is the distributional limit of AoI caused by asynchronous computing modeled as parallel point processes? In other words, what is the connection between computing events in continuous time and the effect on discrete algorithms?

1.4 Contributions and thesis structure

We will now summarize the contributions to the problems and questions raised in Section 1.2 and Section 1.3. We will then conclude the introduction with an outline of the thesis, the recurring chapter structure, and the structural overview figure that illustrates the connections between the chapters.

1.4.1 Main results

Distributed BMT under unbounded information delays

In (Redder, Ramaswamy, and Karl 2023), we developed the anticipated distributed Borkar-Meyn theorem. We studied iteration (1.5) without assuming stability and without assuming a bounded drift function. Compared to the BMT, this analysis requires crucial changes and an altered line of argument to handle the SA errors caused by AoI. As one can expect from (1.7), it is required that the SA stepsize accumulated over intervals with AoI length $\tau_{ij}(n)$ converges to zero almost surely. This condition, which can be stated as

$$\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) \to 0 \text{ a.s.}, \tag{1.12}$$

is the key sufficient condition to prove the distributed BMT and to guarantee almost sure convergence of the distributed SA iteration. The condition relates asymptotic AoI growth and the SA stepsize decay. The distributed BMT Theorem 2.1 then states that (1.5) is stable almost surely under the assumptions of the traditional BMT and (1.12). The key insight enabling the distributed BMT is a careful analysis of the SA errors caused by AoI and the observation that these errors satisfy recursive inequalities in quadratic mean and norm. To evaluate these inequalities,

we propose a new Gronwall-type inequality (Lemma 2.3) to bound iterations that satisfy linear recurrence inequalities with a time-varying lower limit. This inequality may be of independent interest. In summary, this answers the first part of (Q1), which is presented in Chapter 2, but leaves it open to satisfy (1.12) based on AoI properties. Furthermore, it enables an answer to the first part of (Q2).

As an application of the developed stability theory for distributed stochastic approximations, we consider ASGD. Based on the established distributed BMT theorem, we derive conditions for stability and convergence. Further, we analyze the AoI-dependent, almost-sure convergence rate for ASGD, which provides an answer to the second part of (Q2), presented in Chapter 3.

BMT for SA with momentum

As an important byproduct of the distributed BMT, we observed that the principles used to prove the distributed BMT can be used to prove the BMT for SA with momentum. Consider the following SA iteration with Polyak's heavy ball momentum:

$$\begin{aligned} x_{n+1} &= x_n + a(n)m_k, \\ m_k &= \beta m_{k-1} + (1-\beta)g(x_k), \end{aligned} \tag{1.13}$$

where $m_{-1} = 0$, $\beta \in [0,1)$ and $g(x_k) \coloneqq h(x_k) + M_{k+1}$ with drift h and Martingale noise M_{k+1} as before. This iteration has been extensively studied for stochastic gradient descent (SGD) with momentum, but generally only for specific SA iterations or linear SA iterations, whereby a momentum parameter $\beta_n \nearrow 1$ has been chosen.

We observed that (1.13) can be studied by splitting the moving average of past drift terms into "new" and "old" drift terms. Specifically, (1.13) can be written in moving-average form as $x_{n+1} = x_n + a(n)(1-\beta)\left[\sum_{i=1}^{n} \beta^{n-i} g(x_i)\right]$. Then, define a deterministic AoI sequence $\tau(n) \coloneqq \lceil \frac{n}{\sum_{k=0}^{n} a(k)} \rceil$ and split the moving average as follows:

$$x_{n+1} = x_n + a(n)(1-\beta)\left[\sum_{i=n-\tau(n)}^{n} \beta^{n-i} g(x_i)\right] + a(n)(1-\beta)\left[\sum_{i=1}^{n-\tau(n)-1} \beta^{n-i} g(x_i)\right]. \tag{1.14}$$

Under standard assumptions for the drift h and the martingale difference noise M_{n+1}, we will show that the second summation, which averages "old" drifts, is in $o(1)$. Now observe the similarity between the first summation in (1.14) and (1.12). Because of this structural similarity, iteration (1.14) can be studied as an iteration affected by AoI along the lines of the distributed BMT. Specifically, we will conclude that the BMT also holds for heavy-ball stochastic approximations (1.13) and we can therefore provide sufficient conditions for stability of SAs with heavy-ball momentum (Theorem 2.14). This is a new result for general heavy-ball SA iteration fixed momentum parameter and an application of (Q1), which is presented at the end of Chapter 2.

Convergence of distributed actor-critic MARL with aged data

As a specialized distributed SA setting, we studied the convergence of deep multi-agent reinforcement learning over communication networks (Redder, Ramaswamy, and Karl 2022c; Redder, Ramaswamy, and Karl 2022a). The considered setting is a D agent Markov game, illustrated in Figure 1.6.

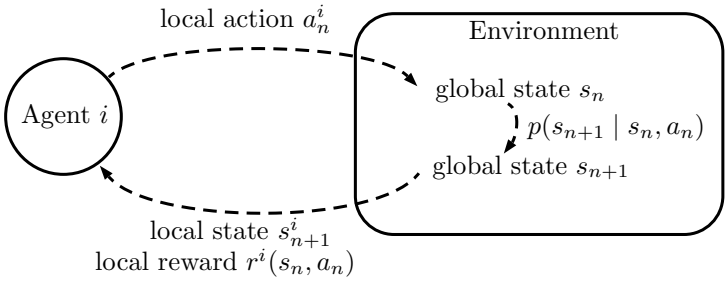

Figure 1.6: Agent interaction with Markov game over discrete time n.

The goal is to learn distributed policies in a decentralized manner using information that is solely communicated between the agents over a communication network, inducing AoI and DataAoI. For this setting, we present 3DPG, an online, fully distributed, multi-agent, actor-critic learning algorithm for networked MAS with continuous decision spaces. Notably, we formulate AoI assumptions that formalize how old information used by the 3DPG agents can be: The DataAoI should not be asymptotically "too large" relative to the used stepsize sequence to guarantee that the obtained multi-agent, actor-critic policy induces that the agent's accumulated experiences constitute stationary distributions over the state Markov process. This property is formally proved as part of the convergence analysis in Chapter 4 and thus provides an answer to (Q3) for the distributed multi-agent learning setting.

To study the convergence of 3DPG, we use recent asymptotic analyses of Deep Q-Learning under mild assumptions (Ramaswamy and Hullermeier 2021), but assuming stability. We show that 3DPG converges to a local stationary point of a Markov game, which, for a special case, implies that the agents converge to a local Nash equilibrium of the Markov Game for policies parameterized by a linear combination of non-linear features. To verify the AoI conditions in the so-far announced results, we developed fundamental results on AoI as a stochastic process.

AoI modeling and deterministic growth bounds

The sawtooth-like nature of AoI sequences in Figures 1.3 and 1.4 may suggest that all AoI processes have a unit growth property, i.e., that $\tau(n+1) \leq \tau(n) + 1$. However, AoI, due to asynchronous computing as introduced in Section 1.3.2, can clearly grow arbitrarily from n to $n+1$ if a single worker is particularly slow. Therefore, AoI sequences with a unit-growth

property will fall into the soon-to-be defined class of *simple AoI processes*; see Definition 5.1.1 in Chapter 5. AoI processes are then defined as processes taking values of one of many simple AoI processes, which was already briefly outlined in Footnote 2. At every time step n, an AoI process will satisfy $\tau(n) = \tau_k(n)$ for some simple AoI processes $\tau_k(n)$ for $k \in \mathcal{K}$, where $\mathcal{K}$ represents a set of simple AoI processes. The idea behind this definition is that information may flow over multiple paths from a source to a monitor. In the asynchronous computing setting, the paths are the (potentially time-varying) set of workers that compute updates, which age as the workers compute them.

Based on the structural definition of an AoI process, we will infer a fundamental growth property, versions of which were discussed in (Redder, Ramaswamy, and Karl 2022a; Redder, Ramaswamy, and Karl 2023). Let $\tau(n)$ be an AoI process with simple AoI process family $\mathcal{K}$, such that each $\tau_k(n)$ is stochastically dominated by a random variable $\overline{\tau}_k$ with $\mathbb{E}\left[\overline{\tau}_k^{p_k}\right] < \infty$ for some $p_k > 0$, then for all $\varepsilon > 0$

$$\mathbb{P}\left(\tau(n) > \varepsilon n^{\frac{1}{\max\{1,p\}}} \text{ i.o.}\right) = 0, \tag{1.15}$$

with $p \coloneqq \min_k\{p_k\}$. This property enables that if an AoI process has dominating random variables with any moment bounds $p_k > 0$, then $a(n) \in \mathcal{O}\left(\frac{1}{n}\right)$ will be sufficient to guarantee that the key property (1.12) holds, thus enabling the established core stability and convergence theorems. These results answer the second part of (Q1) as well as (Q2), (Q5), and (Q6), and are presented in Chapter 5

AoI arising from strongly mixing event processes

It is now left to characterize when such moment bounds to guarantee (1.15) are satisfied.

The deterministic growth bound (1.15) given AoI dominating random variables with some bounded moment provides the first tool for verifiable stability and convergence conditions. Next, we are interested in the existence of dominating random variables with prescribed moment bounds. The fundamental AoI process (1.8) provides the first tool. We describe the event sequences $A(n)$ using the notion of strong mixing. In (Redder, Ramaswamy, and Karl 2022b), we showed that a dominating variable with bounded p-th moment exists for a fundamental AoI process whenever $A(n)$ is strongly mixing with mixing rate $\alpha(A, n)$ such that $\sum_{n\geq 0} n^{p-1}\alpha(A, n) < \infty$. In addition, we show that the fundamental AoI process is itself strongly mixing with almost the same rate as $A(n)$, which enables a strong law for AoI processes – a result of independent interest. These results answer (Q7) and provide the first set of verifiable AoI conditions to answer (Q4) and the second part (Q1). The results are presented in Chapter 6.

AoI arising from asynchronous computing modeled as parallel point processes

We study the AoI arising from asynchronous computing. We model the processing time of parallel workers as parallel point processes. The resulting AoI sequence will then be an AoI process as introduced above, with the simple AoI processes of the k-th worker in continuous time given by

$$\sum_{j \neq k} \left(N_j(t) - N_j(t - B_k(t)) \right), \tag{1.16}$$

where $N_j(t)$ is the point process associated with the j-th worker and $B_k(t)$ the backward recurrence time (Definition 0.0.12) of the k-th worker. In other words, (1.16) simply counts the number of updates from other workers, from the last update of worker k until time t. Here, the main contribution is the weak convergence analysis of the resulting AoI process and the development of sharp moment bounds, which lead to dominating random variables for the AoI process. The results were developed in (Redder 2023) for renewal processes and, in general, apply to point processes that asymptotically converge to stationary increments. Finally, we discuss that the aforementioned event processes may also replace the point process in this asynchronous computing model. With these results, we answer (Q8) and provide the second set of verifiable AoI conditions to answer (Q4) and the second part (Q1). The results are presented in Chapter 7

Decentralized learning in mobile wireless networks

Finally, we discuss how the developed theory applies to learning and optimization when nodes communicate over wireless networks with interference. Apparently, the state of a network channel at successive time steps can be highly dependent. For example, one can consider that the channel state is good if the signal-to-interference-plus-noise ratio (SINR) of a received signal is above a certain threshold, thus guaranteeing successful communication (Tse and Viswanath 2005). The representative model studies the SINR between every pair of agents in a MAS combined with a class of AoI-aware medium access control policies. The core result is that under "sufficiently recurring" Markovian mobile dynamics and carefully chosen medium access policies, the SINR between every pair of agents will be strongly mixing. Hence, the new results on the existence of AoI dominating random variables and AoI growth bounds apply, which thus guarantees stability and convergence for decentralized learning and optimization methods. This application thus combines most of the developed results and provides verifiable conditions to ensure properties of distributed iterations. Regarding the raised questions, this chapter provides a specialized answer to (Q4) and (Q7). The results are presented in Chapter 8.

1.4.2 Thesis structure

The thesis is split into two main parts. Part I, corresponding to chapters Chapters 2 to 4 focuses on distributed SA iterations. In Chapter 2, we study the stability of distributed SA iterations and present the distributed BMT and the BMT for SA with momentum. From this, we conclude in Chapter 3 stability and convergence statements for DASGD and prove the AoI-dependent, almost-sure convergence rate. Afterward, we present 3DPG, a distributed multi-agent algorithm for Markov Games in Chapter 4.

Part II, corresponding to Chapters 5 to 9 then presents the established results on AoI. Chapter 5 discusses modeling stochastic information delays and deterministic AoI growth bounds. Next, the AoI arising from strongly mixing event processes is discussed in Chapter 6. Then, AoI processes arising from distributed computing modeled as parallel point processes are studied in Chapter 7. Finally, AoI-dependent network scheduling policies that preserve mixing properties of SINR in mobile wireless communication are presented in Chapter 8.

In addition, Chapter 9 provides a number of supporting numerical experiments. The code for all experiments is available on `https://github.com/aredder/`. Apart from the numerical experiment chapter, all other chapters have the following structure: After a brief introduction of the problem and required background, the system model and assumptions are presented, and the main results of the chapter are derived as theorems and lemmas. Afterward, each chapter contains a section with discussion and related work. Finally, every chapter closes with a chapter-specific appendix containing all proofs of the chapter omitted in the chapter's main text for better readability. Overall, the thesis concludes with discussions, future work, and an appendix on analysis (A.1) and probability (A.2) for easy reference.

Structure overview

Below, we present a graphical structural overview of the connections between the chapters.

Additionally, we present Table 1.1 to quickly access the chapters relevant to the raised questions in the introduction.

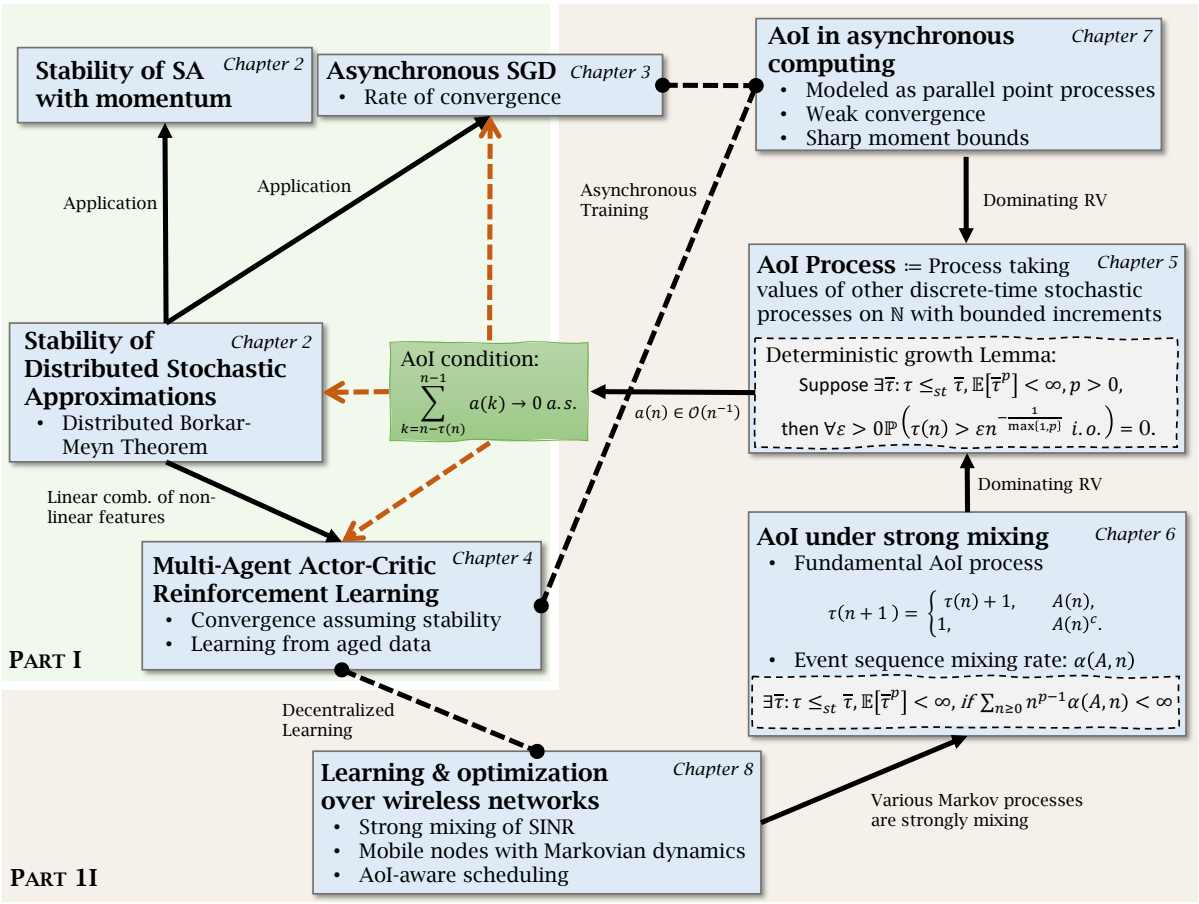

Figure 1.7: Structure overview: Black arrows indicate an implication, dashed orange arrows indicate a sufficient condition, and dashed lines connect an algorithm framework and a communication/AoI model.

	(Q1)	(Q2)	(Q3)	(Q4)	(Q5)	(Q6)	(Q7)	(Q8)
Chapter 2	✓							
Chapter 3		✓						
Chapter 4			✓					
Chapter 5	✓	✓			✓	✓		
Chapter 6	✓			✓			✓	
Chapter 7	✓			✓				✓
Chapter 8				✓			✓	

Table 1.1: Assignment of chapters that address parts of the raised questions.

Part I: Distributed Asynchronous Stochastic Approximation Algorithms

Chapter 2

Stability of Distributed Asynchronous Stochastic Approximations

In this chapter, we derive sufficient conditions for the stability and convergence of distributed SAs (1.5) in the presence of large unbounded stochastic AoI. We generalize the Borkar-Meyn stability theorem from centralized SAs to distributed SAs. Further, we prove the BMT for SA with heavy ball momentum. The results are based on (Redder, Ramaswamy, and Karl 2023). Using AoI properties to be derived in Part II, we can then conclude with assumptions showing that iteration (1.5) with drift $h(\cdot)$ is stable and converges almost surely to a compact connected invariant set of the ODE $\dot{x}(t) = h(x(t))$ provided merely that there exists *an arbitrary* $p > 0$, such that $\sup_{n\geq 0} \mathbb{E}\left[\tau_{ij}^p(n)\right] < \infty$. As mentioned before, prior to this contribution, this was only known if either (1.5) is assumed to be stable almost surely and in addition that the above condition holds for at least some $p > 1$ (V. Borkar 2022, Section 6), or if the AoI variables $\tau_{ij}(n)$ are bounded (Shalabh Bhatnagar 2011).

2.1 Assumption, main statements and preliminaries

Recall that we consider the following iteration:

$$x_{n+1}^i = x_n^i + a(n)\left[h^i(x^1_{n-\tau_{i1}(n)}, \dots, x^D_{n-\tau_{iD}(n)}) + M_{n+1}^i\right], \quad n \geq 0, \quad (1 \leq i \leq D), \tag{2.1}$$

with $x_n \coloneqq (x_n^1, \dots x_n^D)$ in $\mathbb{R}^d \coloneqq \mathbb{R}^{d_1} \times \dots \times \mathbb{R}^{d_D}$, where $\mathbb{R}^d$ is equipped with some norm $\|\cdot\|$, and each $\mathbb{R}^{d_i}$ is equipped with the induced norm on the coordinate subspace. As before, $\tau_{ij}(n)$ is the AoI random variable that incurs since iteration i uses the iteration value of iteration j from time $n - \tau_{ij}(n)$ to evaluate its local drift $h^i : \mathbb{R}^d \to \mathbb{R}^{d_i}$ at time n. Further, M_{n+1}^i is the local Martingale difference noise sequence, and $a(n)$ is the stepsize used by every iteration.

Define the local errors due to AoI at step n as

$$e_n^i \coloneqq h^i(x^1_{n-\tau_{i1}(n)}, \ldots, x^D_{n-\tau_{iD}(n)}) - h^i(x^1_n, \ldots, x^D_n). \tag{2.2}$$

When $e_n^i \in o(1)$ almost surely, then by inspection, we expect that under suitable assumptions on $a(n)$, h and M_{n+1}^i iteration (2.1) will track solutions to the dynamical system

$$\dot{x}(t) = h(x(t)). \tag{2.3}$$

The standard regularity assumption to ensure that the ODE (2.3) is well-posed (V. Borkar 2022, App. B) is that h is Lipschitz continuous:

Assumption 2.1.1. *$h^i : \mathbb{R}^d \to \mathbb{R}^{d_i}$ is the i-th component of a Lipschitz-continuous map $h : \mathbb{R}^d \to \mathbb{R}^d$ with Lipschitz constant $L > 0$.*

The Lipschitz condition plays a crucial role in establishing the stability and convergence of (2.1) in the presence of the errors e_n^i. In addition, we assume that rescaled versions of the ODE (2.3) converge to an ODE with a globally asymptotically stable equilibrium.

Assumption 2.1.2. *The functions $h_c(x) \coloneqq \frac{h(cx)}{c}, c \geq 1$, $x \in \mathbb{R}^d$, satisfy $h_c(x) \to h_\infty(x)$ pointwise as $c \to \infty$ for some $h_\infty \in \mathcal{C}(\mathbb{R}^d)$. Furthermore, the ODE $\dot{x}(t) = h_\infty(x(t))$ has the origin as its globally asymptotically stable equilibrium.*

This is the BMT stability condition recalled from Condition 1 in Chapter 1. As mentioned before, the main contribution is extending the BMT to a distributed setting with unbounded stochastic delays. Inspired by the proof of the distributed BMT, we will also present a weaker version of Assumption 2.1.2 in Section 2.2.4 that also applies to the traditional BMT.

We make the following assumption for the additive noise terms M_{n+1}^i.

Assumption 2.1.3. *$M_{n+1}^i \in \mathbb{R}^{d_i}$ is the i-th component of a martingale difference noise process $\{M_n\}$ with respect to the filtration $\mathcal{F}_n \coloneqq \sigma\left(x_0, M_1, \ldots M_n, \tau_{ij}(0), \ldots \tau_{ij}(n), 1 \leq i, j \leq D\right), n \geq 0$:*

1) $\{M_{n+1}\}_{n\geq 1}$ is a square integrable sequence.
2) $\mathbb{E}\left[\|M_{n+1}^i\|^2 \mid \mathcal{F}_n\right] \leq K^2\left(1 + \|(x^1_{n-\tau_{i1}(n)}, \ldots, x^D_{n-\tau_{iD}(n)})\|^2\right)$ for some $K > 0$.

Assumption 2.1.3 bounds the conditional second moment of the martingale noise component at time $n \geq 0$ based on the associated iteration values in (2.1); a weaker version will be discussed in Section 2.2.4. Finally, we state the assumptions for the stepsize sequence $a(n)$ and the AoI sequences $\tau_{ij}(n)$. As mentioned in the introduction, the AoI sequences can be the consequence of various transport phenomena resulting in the effective use of aged information $x^j_{n-\tau_{ij}(n)}$ when evaluating the recursion (2.1).

Assumption 2.1.4. *The stepsize $a(n)$ is not summable but square summable, i.e., $\sum_{n\geq 0} a(n) = \infty$ and $\sum_{n\geq 0} a(n)^2 < \infty$.*

Assumption 2.1.5. *For all i, j, the stepsize $a(n)$ and the AoI sequences $\tau_{ij}(n)$ guarantee that*

$$\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) \to 0 \text{ a.s..}$$

Assumption 2.1.5, which corresponds to the announced key property (1.12) for the stability and convergence of distributed, describes the trade-off between the stepsize sequence and the AoI sequences. A faster stepsize decay allows the AoI sequences to be larger asymptotically. The details of the stability proof will show the importance of this assumption in conjunction with Lipschitz continuity for the stability of distributed SAs. Notably, it turns out that the almost sure convergence in Assumption 2.1.5 reaches deeper into the stability analysis than one might initially expect since the convergence is almost uniformly.

Theorem 2.1 (Distributed Borkar-Meyn Theorem)**.** *Under Assumption 2.1.1-2.1.5, iteration (2.1) is stable almost surely, i.e., $\sup_n \|x_n\| < \infty$ a.s..*

Corollary 2.2. *Under Assumption 2.1.1-2.1.5 iteration (2.1) converges almost surely to a potential sample path-dependent compact connected internally chain transitive invariant set of the ODE (2.3).*

For the terminology used to state Corollary 2.2, see V. Borkar (2022, Sec. 2.1). We do not explain it here, as it is unnecessary for the stability analysis. The most important consequence of Corollary 2.2 is that when the only invariant sets (Definition 0.0.3) are isolated equilibrium points (Definition 0.0.4), then x_n converges almost surely to a potential sample path-dependent equilibrium point (V. Borkar 2022, Sec. 2.2).

2.1.1 Traditional Borkar-Meyn theorem

The proof of Theorem 2.1 is inspired by the traditional BMT but requires crucial changes to handle the a priori non-vanishing additive drift error e_n^i due to AoI. To better understand the required changes, we first sketch the proof of the traditional BMT, i.e., we consider iteration x_n in (2.1) with $\tau_{ij}(n) = 0$ for all $n \geq 0$.

As the first step, the BMT proof creates a piecewise linear interpolated trajectory $\overline{x}(t)$ from the iteration x_n. Then, the time axis $[0, \infty)$ is seperated into concatenated time intervals $[T_m, T_{m+1}]$ of length approximately $T > 0$. Finally, a rescaled trajectory $\hat{x}(t)$ is created by dividing $x(t)$ over each $[T_m, T_{m+1}]$ by $\|x(T_m)\|$, i.e., each $x(T_m)$ is scaled to the unit ball. *The second step* is to show that the rescaled trajectory $\hat{x}(t)$ is stable almost surely. Then a stochastic approximation argument shows that $\hat{x}(t)$ tracks solutions to scaled ODEs with drift $h_c(\cdot)$, $c \in [1, \infty]$, from Assumption 2.1.2. *The final step* is to verify the stability by contradiction. Assuming that x_n is unstable, a subsequence of scaling factors $\|x(T_m)\|$ diverging to infinity will exist. A stochastic

approximation argument then leads to corresponding rescaled segments of $\hat{x}(t)$ that asymptotically track the limiting ODE from Assumption 2.1.2 with drift $h_\infty(\cdot)$. Since this limiting ODE is globally asymptotically stable to the origin, the aforementioned rescaled segments eventually drift toward a neighborhood of the origin. This leads to a contradiction since $\|x(T_m)\|$ diverges to infinity. We will now begin with preliminaries for the proof of the distributed BMT.

2.1.2 Recursive structure of stochastic approximation errors caused by AoI

Rewrite the main iteration (2.1) as

$$x_{n+1} = x_n + a(n)\left[h(x_n) + e_n + M_{n+1}\right], \tag{2.4}$$

for $n \geq 0$, with additive drift error $e_n = (e_n^1, \ldots, e_n^D)$ composed of local drift errors e_n^i as defined in (2.2). We shall impose the natural condition that iteration (2.1) (thence (2.4)) starts from a prescribed x_0 with $(\mathbb{E}\left[\|x_0\|^2\right])^{\frac{1}{2}} < \infty$. Assumption 2.1.1 then implies the linear growth of $h(\cdot)$:

$$\|h(x)\| \leq K(1 + \|x\|) \tag{2.5}$$

for all $x \in \mathbb{R}^d$ for some $K > 0$ depending on x_0. To simplify the presentation, we assume without loss of generality that the same constant K holds for both (2.5) and the inequality in Assumption 2.1.3.

Using the Lipschitz-continuity of h (Assumption 2.1.1), the norm of the local drift errors satisfy

$$\|e_n^i\| \leq L \sum_{j=1}^{D} \|x_n^j - x_{n-\tau_{ij}(n)}^j\|. \tag{2.6}$$

Next, using a telescoping sum and the triangular inequality, it follows that

$$\|x_n^j - x_{n-\tau_{ij}(n)}^j\| \leq \sum_{k=n-\tau_{ij}(n)}^{n-1} \|x_{k+1}^j - x_k^j\| \tag{2.7}$$

$$= \sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) \left(\|h^j(x_{k-\tau_{j1}(k)}^1, \ldots, x_{k-\tau_{jD}(k)}^D) + M_k^j\| \right), \tag{2.8}$$

where the second step uses iteration (2.1). Note that whenever $\tau_{ij}(n) = 0$, the sums on the right-hand side are empty and thus equal to zero. From the last inequality, one can imagine that $\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) \to 0$ a.s. (Assumption 2.1.5) is a sufficient condition to prove the stability of (2.1). This is immediate from the BMT provided that 1. the drift h is bounded almost surely and 2. the noise M_{n+1} is bounded almost surely. We do not make these assumptions.

Using the linear growth of h, (2.5), we conclude that

$$\begin{aligned} \|x_n^j - x_{n-\tau_{ij}(n)}^j\| \leq K \sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) \left(1 + \|x_k\| + \sum_{l=1}^{D} \|x_k^l - x_{k-\tau_{jl}(k)}^l\| \right) \\ + \| \sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) M_k^j \|. \end{aligned} \tag{2.9}$$

In the following, we will motivate some key steps from inequality (2.9).

2.1.3 Creation of rescaled trajectories

As the first step, along the lines of the traditional BMT, we create a piecewise linear interpolated trajectory $\overline{x}(t)$ and a rescaled trajectory $\hat{x}(t)$ from x_k. Divide the time axis $[0,\infty)$ using the stepsize $a(n)$ as follows. Define time instants

$$t(0) \coloneqq 0, \quad t(n) \coloneqq \sum_{i=1}^{n-1} a(i), \qquad \text{for all } n \geq 1. \tag{2.10}$$

Now define an interpolated trajectory $\overline{x}(t)$, by setting $\overline{x}(t(n)) \coloneqq x_n$, $n \geq 0$ and define $\overline{x}(t)$ for all other points $t \in [0,\infty)$ by linear interpolation. Fix $T > 0$, then split the time axis into approximately T-length intervals with initial time steps

$$T_0 = 0, \quad T_{m+1} \coloneqq \min\{t(n) : t(n) \geq T_m + T\}. \tag{2.11}$$

In contrast to the traditional BMT, we now require a different, larger, rescaling sequence to create $\hat{x}(t)$. From a physical, dynamical systems perspective, we have to scale the iteration more in the presence of AoI, since otherwise older iteration values that are not sufficiently scaled lead to scaled drift errors that cause $\hat{x}(t)$ to be unstable. Technically, this can be seen from inequality (2.9). If we were to use the rescaling sequence $\|x(T_m)\|$ as for the BMT, then (2.9) shows that a bound for the local drift error e_n (thence for x_n) depends on x_k with $k \in \{n-1, \dots n-\tau_{ij}(n)\}$. The problem is that these x_k will be associated with different T-length intervals $[T_m, T_{m+1}]$ than x_n, whenever the AoI $\tau_{ij}(n)$ is large. Thus scaling both sides of (2.9) with the scaling factor associated with x_n does not lead to variables on the right-hand side of (2.9) that can be bounded by the rescaled trajectory $\hat{x}(t)$.

To solve the problem of the original scaling sequence, we propose that the interpolated trajectory is scaled over every T-length interval by the larger scaling sequence

$$s(m) \coloneqq \max\{\sup_{l \leq m} \|\overline{x}(T_l)\|, 1\}, m \geq 0. \tag{2.12}$$

Remark 2.1.1. *From a dynamical systems perspective, the traditional BMT studies a projected version of x_n, where after every T-length interval in continuous time, the iteration is projected to the unit sphere. Instead, we study a projected version of x_n, where after every T-length interval in continuous time, the iteration is projected to a point on the unit ball. This is done by adapting the scaling value to project the iteration: whenever the projected iteration would be outside the unit ball with the previous scaling value, the value is increased such that the projected iteration is on the unit sphere.*

The crucial point of constructing $s(m)$ is that $s(m)$ is monotonically increasing. The rescaled version of $\overline{x}(t)$ is then defined as

$$\hat{x}(t) \coloneqq \frac{\overline{x}(t)}{s(m)}, \qquad t \in [T_m, T_{m+1}), \tag{2.13}$$

In addition, define the rescaled drift error sequence and the rescaled martingale noise sequence as follows by

$$\hat{e}_n \coloneqq \frac{e_n}{s(m)} \text{ and } \hat{M}_{n+1} \coloneqq \frac{M_{n+1}}{s(m)}, \tag{2.14}$$

respectively, for $n \in [T_m, T_{m+1}), m \geq 0$. Finally, define the accumulated rescaled noise sequence by

$$\hat{\zeta}_n \coloneqq \sum_{k=1}^{n-1} a(k)\hat{M}_{k+1}. \tag{2.15}$$

The second step is to show that $\hat{x}(t)$ is stable almost surely, for which we proceed as follows:

1) We prove L_2 bounds for $\hat{x}(t)$. Specifically, we show that $\sup_t \mathbb{E}\left[\|\hat{x}(t)\|^2 \mid E_z\right] < \infty$ for an increasing sequence of events E_z (Lemma 2.5).
2) We show that $\hat{\zeta}_n$ (2.15) is convergent almost surely (Lemma 2.6).
3) We show that $\sup_{t\geq 0}\|\hat{x}(t)\| < \infty$ a.s. and as a corollary that the rescaled error vanishes, i.e. $\|\hat{e}_n\| \to 0$ a.s. (Lemma 2.8 and Corollary 2.9, respectively).

While straightforward in the BMT, these steps are much more involved in the presence of the stochastic drift errors e_n^i due to AoI. Equation (2.9) indicates that we need to bound $\mathbb{E}\left[\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)\|x_k^l - x_{k-\tau_{jl}(k)}^l\|^2\right]$ to show that $\hat{x}(t)$ is bounded in L_2. However, this is an expected value of a random number of random variables. To circumvent this, we will use that $\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)$ converges almost uniformly. We can, therefore, work with deterministic upper bounds $\Delta(n)$ for every $\tau_{ij}(n)$ on an increasing sequence of probability subspaces. It will then follow that

$$\mathbb{E}\left[\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)\|x_k^l - x_{k-\tau_{jl}(k)}^l\|^2\right] \leq \sum_{k=n-\Delta(n)}^{n-1} a(k)\mathbb{E}\left[\|x_k^l - x_{k-\tau_{jl}(k)}^l\|^2\right], \tag{2.16}$$

and we will show that the required L_2 bound holds on an increasing sequence of probability subspaces.

Finally, (2.9) also shows that a bound for $\|x_n^j - x_{n-\tau_{ij}(n)}^j\|$ depends on all $\|x_k^l - x_{k-\tau_{jl}(n)}^l\|$ for all $1 \leq l \leq D$ and $k \in \{n-1, \dots n-\tau_{ij}(n)\}$. Here, the critical observation is that if we sum up both sides of (2.9) over all $1 \leq i, j \leq D$ then a recursive inequality in the variable $\sum_{i,j=1}^D \|x_n^j - x_{n-\tau_{ij}(n)}^j\|$ arises. Indeed, this recursive structure arises in L_2 and in norm, i.e., in steps 1 and 3 mentioned above, to show that $\hat{x}(t)$ is stable. To evaluate these recursive structures, we present a new Gronwall-type inequality.

2.1.4 A discrete Gronwall-type inequality for varying lower time-horizons

We present the Gronwall-type inequality in greater generality than necessary for the analysis since it may be of independent interest.

Lemma 2.3. *Let $\{y_n\}$, $\{a_n\}$, $\{b_n\}$, $\{c_n\}$, $\{\Delta_n\}$ be non-negative sequences, with $\{b_n\}$ bounded and $\{c_n\}$ monotonically increasing and $L > 0$ be scalar, such that for all $n \geq 0$,*

$$y_n \leq b_n c_n + L \sum_{k=n-\Delta_n}^{n-1} a_k y_k, \tag{2.17}$$

$$N \coloneqq \inf\{N \in \mathbb{N} : L \sum_{k=n-\Delta_n}^{n-1} a_k \leq \frac{e^{Lt(N)} - 1}{e^{Lt(N)}} \text{ for all } n \geq N\} < \infty. \tag{2.18}$$

where $t(0) \coloneqq 0, t(n) \coloneqq \sum_{i=0}^{n-1} a(i)$ for $n \geq 1$. Then

$$y_n \leq c_n \left(b_n + (\sup_{k \geq 0} b_k) L e^{Lt(N)} \left(\sum_{k=n-\Delta_n}^{n-1} a_k \right) \right). \tag{2.19}$$

The nature of Lemma 2.3 is that $\sum_{k=n-\Delta_n}^{n-1} a_k$ leads to weighted averaging of the y_k sequence, such that asymptotically the right-hand side of (2.17) is neglectable, when $\sum_{k=n-\Delta_n}^{n-1} a_k \to 0$. We will now apply Lemma 2.3 in the following subsections to show that $\hat{x}(t)$ is stable.

2.2 Distributed Borkar-Meyn Theorem

We will now present the core steps to prove the distributed BMT.

2.2.1 Recursive L_2 structure and L_2 Bounds

It will become useful to have a function $m(n)$ that selects for each discrete time $n \geq 0$ the corresponding interval $[T_{m(n)}, T_{m(n)+1})$ in continuous time. In other words, $m(n)$ is the largest interval index m, such that $T_m \leq \sum_{i=0}^{n-1} a(i)$.

Recall the filtration $\mathcal{F}_n$ defined in Assumption 2.1.3. We have that $\sigma(x_1, \ldots, x_n) \subset \mathcal{F}_n$ and by construction of the time intervals in (2.11), we have that $m(n) \leq n$. It follows that $s(m(n)) \in \mathcal{F}_n$,

with the scaling sequence $s(\cdot)$ as defined in (2.12). Assumption 2.1.3 therefore leads to

$$\mathbb{E}\left[\|\hat{M}^i_{n+1}\|^2 \mid \mathcal{F}_n\right] = \mathbb{E}\left[\frac{\|M^i_{n+1}\|^2}{s(m(n))^2} \mid \mathcal{F}_n\right] \tag{2.20}$$

$$= \frac{\mathbb{E}\left[\|M^i_{n+1}\|^2 \mid \mathcal{F}_n\right]}{s(m(n))^2} \tag{2.21}$$

$$\leq \frac{K^2\left(1 + \|(x^1_{n-\tau_{i1}(n)}, \dots, x^D_{n-\tau_{iD}(n)})\|^2\right)}{s(m(n))^2} \tag{2.22}$$

$$\leq \frac{K^2\left(1 + \|x_n\|^2 + \sum_{j=1}^D \|x^j_n - x^j_{n-\tau_{ij}(n)}\|^2\right)}{s(m(n))^2} \tag{2.23}$$

$$\leq K^2\left(1 + \|\hat{x}(t(n)\|^2 + \sum_{j=1}^D \left(\frac{\|x^j_n - x^j_{n-\tau_{ij}(n)}\|}{s(m(n))}\right)^2\right) \tag{2.24}$$

By taking the expected value and the square root of the last inequality, we arrive at

$$\mathbb{E}\left[\|\hat{M}^i_{n+1}\|^2\right]^{\frac{1}{2}} \leq K\left(1 + \mathbb{E}\left[\|\hat{x}(t(n)\|^2\right]^{\frac{1}{2}} + \sum_{j=1}^D \mathbb{E}\left[\left(\frac{\|x^j_n - x^j_{n-\tau_{ij}(n)}\|}{s(m(n))}\right)^2\right]^{\frac{1}{2}}\right) \tag{2.25}$$

for all $n \geq 0$, where we used that the square root of a sum is bounded by the sum of the square roots of its terms. In addition, using (2.6), we can bound the rescaled additive drift errors in L_2:

$$\mathbb{E}\left[\|\hat{e}^i_n\|^2\right]^{\frac{1}{2}} \leq L\sum_{j=1}^D \mathbb{E}\left[\left(\frac{\|x^j_n - x^j_{n-\tau_{ij}(n)}\|}{s(m(n)}\right)^2\right]^{\frac{1}{2}}. \tag{2.26}$$

Next, divide both sides of the rewritten main iteration (2.4) by $s(m(n))$, take the norm on both sides, and use the linear growth of the iteration drift (2.5). Then,

$$\|\hat{x}(t(n+1))\| \leq \|\hat{x}(t(n))\|(1 + a(n)K) + a(n)(1 + \|\hat{e}_n\| + \|\hat{M}_{n+1}\|). \tag{2.27}$$

Finally, take $\mathbb{E}\left[(\cdot)^2\right]^{\frac{1}{2}}$ on both sides above and use (2.25) and (2.26) componentwise to arrive at the following recursive L_2 bound for $\|\hat{x}(t(n))\|$:

$$\mathbb{E}\left[\|\hat{x}(t(n+1))\|^2\right]^{\frac{1}{2}} \leq \mathbb{E}\left[\|\hat{x}(t(n))\|^2\right]^{\frac{1}{2}}(1 + a(n)K_1) + a(n)\left(K_1 + K_2\sum_{i,j=1}^D \mathbb{E}\left[\left(\frac{\|x^j_n - x^j_{n-\tau_{ij}(n)}\|}{s(m(n))}\right)^2\right]^{\frac{1}{2}}\right). \tag{2.28}$$

with $K_1 \coloneqq K(D+1)$ and $K_2 \coloneqq L + K$.

To continue, we have to consider the local errors $x^j_n - x^j_{n-\tau_{ij}(n)}$. Equation (2.9) leads to

$$\|x^j_n - x^j_{n-\tau_{ij}(n)}\| \leq K\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)\left(1 + \|x_k\| + \sum_{l=1}^D \|x^l_k - x^l_{k-\tau_{jl}(k)}\|\right) + \sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)\|M^j_k\|. \tag{2.29}$$

To move forward, we must take the expected value of (2.29). This requires that we take the expected value of a random number of random variables, which is generally difficult without restrictive assumptions (see, e.g., Wald's Lemma in Durrett (2019, Chap. 2)). We observed that we could circumvent this difficulty using almost uniform convergence implied by Egorovs's Theorem (see A.2).

Definition 2.2.1 (Almost uniform convergence)**.** *Let X_n, X be random variables on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Then X_n is said to converge to X almost uniformly if, for every $\varepsilon > 0$, there exists an exceptional set $A \in \mathcal{F}$ with $\mathbb{P}(A) < \varepsilon$ such that X_n converges uniformly to X on the complement $E = \Omega \setminus A$.*

By Assumption 2.1.5, $\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)$ converges almost surely and thus Egorov's theorem implies that $\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)$ converges almost uniformly. Consequently, we can work with deterministic upper bounds for all $\tau_{ij}(n)$ on an increasing sequence of probability subspaces. This is sufficient since the objective is to show that the accumulated rescaled noise sequence $\hat{\zeta}_n$ is convergent almost surely.

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be the underlying probability space, i.e., $(\Omega, \mathcal{F}, \mathbb{P})$ is the common probability space on which the stochastic processes $\{x_n\}_{n\geq 0}$, $\{M_{n+1}\}_{n\geq 0}$ and all $\{\tau_{ij}(n)\}_{n\geq 0}$ are defined. Fix a sequence $\{\varepsilon_z\}_{z\geq 0} \subset (0,1)$ with $\varepsilon_z \to 0$. The almost uniform convergence of $\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)$ thus implies that there are $E_0, E_1, \ldots, E_z, \ldots \in \mathcal{F}$ with $\mathbb{P}(E_z) \geq 1-\varepsilon_z$, such that $\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) \to 0$ uniformly on each E_z. We may assume that $E_0 \subset E_1 \subset E_2 \ldots$ as uniform convergence on finite unions follows from the uniform convergence on the individual sets. It now follows that it is sufficient for the convergence of the accumulated noise iteration $\hat{\zeta}_n$ to show that $\sup_t \mathbb{E}\left[\|\hat{x}(t)\|^2 \mid E_z\right] < \infty$ for every $z \geq 0$, which is discussed in Remark 2.2.1 below. Indeed, we may now also assume without loss of generality that all $\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) \to 0$ uniformly on Ω. However, we prefer to keep the conditioning to avoid confusion. Therefore, define $\mathbb{E}_z[\cdot] := \mathbb{E}[\cdot \mid E_z]$ for all $z \geq 0$.

Remark 2.2.1. *Recall that we need to show that $\mathbb{P}\left(\hat{\zeta}_n \text{ converges}\right) = 1$. Suppose that we can show that $\mathbb{P}\left(\hat{\zeta}_n \text{ converges} \mid E_z\right) = 1$ for all $z \geq 0$. Then, by the construction of the sets E_z, we have $\mathbb{P}\left(E_z \cap \{\hat{\zeta}_n \text{ converges}\}\right) = \mathbb{P}(E_z) \geq 1 - \varepsilon_z$. Using continuity from below, as E_z are increasing and $\varepsilon_z \to 0$ as $z \to \infty$, it follows that $\mathbb{P}\left(\hat{\zeta}_n \text{ converges}\right) = \mathbb{P}\left(\bigcup_{z\geq 0} E_z \cap \{\hat{\zeta}_n \text{ converges}\}\right) = 1$.*

Remark 2.2.2. *All previous L_2 bounds from this subsection also hold for $\mathbb{E}_z[\cdot]$. In particular, inequality (2.25) holds with K replaced by $\frac{K}{\sqrt{\mathbb{P}(E_z)}}$, since*

$$\mathbb{E}_z\left[\|\hat{M}_{n+1}^i\|^2 \mid \mathcal{F}_n\right] \leq \frac{\mathbb{E}\left[\|\hat{M}_{n+1}^i\|^2 \mid \mathcal{F}_n\right]}{\mathbb{P}(E_z)}. \tag{2.30}$$

Then using (2.24) and taking $\mathbb{E}_z[\cdot]$ we arrive at the required inequality.

We will now use the deterministic sequences

$$\Delta_z(n) \coloneqq \sup_{\omega \in E_z} \{\tau_{ij}(n)(\omega) \mid 1 \leq i,j \leq D\}. \tag{2.31}$$

By construction of the events E_z, it follows that $\sum_{k=n-\Delta_z(n)}^{n-1} a(k) \to 0$ as $n \to \infty$ uniformly on E_z. Further, $\tau_{ij}(n) \leq \Delta_z(n)$ on E_z for all $1 \leq i,j \leq D$ and all $n \geq 0$. Hence, Equation (2.29) implies

$$\|x_n^j - x_{n-\tau_{ij}(n)}^j\| \leq K \sum_{k=n-\Delta_z(n)}^{n-1} a(k) \left(1 + \|x_k\| + \sum_{l=1}^{D} \|x_k^l - x_{k-\tau_{jl}(k)}^l\|\right) + \sum_{k=n-\Delta_z(n)}^{n-1} a(k)\|M_k^j\| \tag{2.32}$$

on E_z. Next, divide the above inequality by $s(m(n))$ and use that $s(m(n))$ is by construction monotonically increasing. It follows that

$$\begin{aligned}\frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))} \leq K \sum_{k=n-\Delta_z(n)}^{n-1} a(k) \left(1 + \|\hat{x}(t(k))\| + \sum_{l=1}^{D} \frac{\|x_k^l - x_{k-\tau_{jl}(k)}^l\|}{s(m(k))}\right) \\ + \sum_{k=n-\Delta_z(n)}^{n-1} a(k)\|\hat{M}_k^j\|.\end{aligned} \tag{2.33}$$

Since $\Delta_z(n)$ is deterministic, we can evaluate $\mathbb{E}_z\left[(\cdot)^2\right]^{\frac{1}{2}}$ on both sides of (2.33) and apply the version of (2.25) for $\mathbb{E}_z[\cdot]$. Then, with $K' \coloneqq \left(K + \frac{K}{\sqrt{\mathbb{P}(E_z)}}\right)$, we arrive at

$$\begin{aligned}\mathbb{E}_z\left[\left(\frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))}\right)^2\right]^{\frac{1}{2}} \leq K' \sum_{k=n-\Delta_z(n)}^{n-1} a(k)(1 + \mathbb{E}_z\left[\|\hat{x}(t(k))\|^2\right]^{\frac{1}{2}}) \\ + K' \sum_{k=n-\Delta_z(n)}^{n-1} a(k) \left(\sum_{l=1}^{D} \mathbb{E}_z\left[\left(\frac{\|x_k^l - x_{k-\tau_{jl}(k)}^l\|}{s(m(k))}\right)^2\right]^{\frac{1}{2}}\right).\end{aligned} \tag{2.34}$$

Finally, a summation over all $1 \leq i,j \leq D$ leads to

$$\begin{aligned}\sum_{i,j=1}^{D} \mathbb{E}_z\left[\left(\frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))}\right)^2\right]^{\frac{1}{2}} \leq K'D^2 \sum_{k=n-\Delta_z(n)}^{n-1} a(k) \left(1 + \mathbb{E}_z\left[\|\hat{x}(t(k))\|^2\right]^{\frac{1}{2}}\right) \\ + K'D \sum_{k=n-\Delta_z(n)}^{n-1} a(k) \sum_{i,j=1}^{D} \mathbb{E}_z\left[\left(\frac{\|x_k^j - x_{k-\tau_{ij}(k)}^j\|}{s(m(k))}\right)^2\right]^{\frac{1}{2}}.\end{aligned} \tag{2.35}$$

This is the announced recursive inequality of the AoI error in L_2. To evaluate this inequality, we use Lemma 2.3, the proposed Gronwall-type inequality for varying lower time horizons. The main observation is that the sum $\sum_{k=n-\Delta_z(n)}^{n-1} a(k)$ leads to weighted averaging, such that the left hand of (2.35) can be bounded as a function of $\mathbb{E}_z\left[\|\hat{x}(t(k))\|^2\right]^{\frac{1}{2}}$, $k \leq n-1$. With this, we can finally conclude the announced L_2 bounds for $\|\hat{x}(t)\|$.

Lemma 2.4. *There is a constant $K_3 > 0$, such that*

$$\sum_{i,j=1}^{D} \mathbb{E}_z \left[\left(\frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))} \right)^2 \right]^{\frac{1}{2}} \leq K_3 \left(1 + \sup_{k \leq n-1} \mathbb{E}_z \left[\|\hat{x}(t(k))\|^2 \right]^{\frac{1}{2}} \right) \left(\sum_{k=n-\Delta_z(n)}^{n-1} a(k) \right) \tag{2.36}$$

The combination of Lemma 2.4 and (2.28) then leads to the required conditional L_2 bound.

Lemma 2.5. $\sup_{t \geq 0} \mathbb{E}_z \left[\|\hat{x}(t)\|^2 \right] < \infty$

2.2.2 Stability of the recalled trajectory

With the established L_2 bound for the recalled trajectory, we are now ready to prove its stability. First, we use Lemma 2.5 and the Martingale convergence theorem to show the convergence of the accumulated rescaled noise iteration.

Convergence of the accumulated rescaled martingale noise

By the reasoning in Remark 2.2.1, it is enough to show that $\mathbb{P}\left(\hat{\zeta}_n \text{ converges } \middle| E_z\right) = 1$ for all $z \geq 0$, which follows from Lemma 2.5 and the convergence theorem for square-integrable martingales (see A.2).

Lemma 2.6. $\hat{\zeta}_n$ *converges almost surely.*

We will also need a corollary to Lemma 2.6 that shows that the martingale noise accumulated over intervals with AoI length and scaled by the sequence $s(m(n))$ converges to zero almost surely.

Corollary 2.7. $\frac{1}{s(m(n))} \sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) M_{k+1}^j \to 0$ *a.s.*

Stability of the rescaled trajectory

We are now ready to show that $\hat{x}(t)$ is stable. First, the convergence of the accumulated rescaled martingale noise (Lemma 2.6) implies that it is bounded almost surely, i.e., $\sup_{n \geq 0} \|\hat{\zeta}_n\| < \infty$ almost surely. Using Corollary 2.7 and (2.9), we then arrive at recursive structure for $\sum_{i,j=1}^{D} \frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))}$, which we again evaluate using Lemma 2.3 to obtain that

$$\|\hat{e}_n\| \in \mathcal{O}\left(\left(1 + \sup_{k \leq n-1} \|\hat{x}(t(k))\| \right) \left(\sum_{k=n-\tau(n)}^{n-1} a(k) \right) \right). \tag{2.37}$$

Recall now the functions $h_c(x)$, $c \in [1, \infty]$, defined in Assumption 2.1.2. It is not difficult to verify that:

1) h_c, h_∞ are Lipschitz-continuous with the same Lipschitz constant $L > 0$ as h.
2) $\|h_c(x)\| \leq K(1 + \|x\|)$ for every $c \in [1, \infty]$.

The recalled iteration then satisfies

$$\hat{x}(t(n+1)) = \hat{x}(t(n)) + a(n)\left[h_{s(m)}(\hat{x}(t(n))) + \hat{e}_n + \hat{M}_{n+1}\right]. \tag{2.38}$$

Iterating (2.38) and using (2.37) and Assumption 2.1.5, we can conclude the stability of the rescaled trajectory, which in turn by (2.37) implies that the rescaled error vanishes.

Lemma 2.8. $\sup_{t\geq 0}\|\hat{x}(t)\| < \infty$ *a.s.*

Corollary 2.9. $\|\hat{e}_n\| \to 0$ *a.s.*

2.2.3 Distributed BMT proof

It is left to show that x_n is stable using that $\hat{x}(t)$ is stable and that $h_\infty(\cdot)$ from Assumption 2.1.2 is the drift of an asymptotically stable ODE. The line of argument is new compared to the traditional BMT and was necessary since we defined the scaling sequence as monotonically increasing. Indeed, the line of argument in the traditional BMT does not apply to monotonically increasing scaling sequences. However, as we saw in the previous subsections, the monotonically increasing scaling sequence is required to scale the drift errors due to AoI. We will now present the details to complete the proof. There are two initial steps along the lines of the traditional BMT.

First, we conclude that the rescaled trajectory $\hat{x}(t)$ is a noisy approximation of solutions to ODEs with drift $h_{s(m)}(\cdot)$. For any $m \geq 0$, let $x^m(t)$, $t \in [T_m, T_{m+1}]$, be the unique solution to

$$\dot{x}(t) = h_{s(m)}(x(t)) \tag{2.39}$$

with initial condition $x^m(T_m) = \hat{x}(T_m)$. Recall that the rescaled iteration can be written as given in (2.75). Lemma 2.6 and Corollary 2.9 imply that the rescaled iteration $\hat{x}(t(n))$ has the form of a standard stochastic approximation iteration as in (1.6) with convergent accumulated noise $\hat{\zeta}_n$ and vanishing error $\hat{e}_n$. Further, Lemma 2.8 shows that the rescaled iteration remains bounded almost surely. A stochastic approximation argument; see, e.g., (V. Borkar 2022, Chapter 2, Lemma 1), then shows that $\hat{x}(t)$ is a noisy approximation of solutions to the ODEs (2.39).

Lemma 2.10. $\lim_{m\to\infty} \sup_{t\in[T_m,T_{m+1}]} \|\hat{x}(t) - x^m(t)\| = 0$ *a.s.*

Second, recall that Assumption 2.1.2 ensures the existence of a function $h_\infty(x)$, such that $h_c(x) \to h_\infty(x)$ as $c \to \infty$. By the Lipschitz continuity of h, the Arzelà–Ascoli theorem (see A.1) implies that h_c converges uniformly on compact subsets of $\mathbb{R}^d$ (compactly). Further, h_∞ has the origin as its unique globally asymptotically stable equilibrium. Therefore, solutions to $\dot{x}(t) = h_\infty(x)$ will

eventually reach a neighborhood of the origin after some $T > 0$. Since the functions h_c inherit the Lipschitz-continuity from h, the scaled ODEs have unique solutions for every initialization. For every $c \in [1, \infty]$, let $\phi_c(t, x)$ denote the unique solution of the ODE $\dot{x}(t) = h_c(x(t))$ with $x(0) = x$. The compact convergence of $h_c \to h_\infty$ now guarantees that for large c and initialization on the unit ball, the solutions $\phi_c(t, x)$ will reach a neighborhood of the origin after $T > 0$. This is stated as the following lemma; for details, we refer to (V. Borkar 2022, Chap. 3, Cor. 4.1) or the original BMT paper.

Lemma 2.11. *There exist $c_0 \geq 1$ and $T > 0$ such that for all initial conditions x on the closed unit ball, $\|\phi_c(x, t)\| < \frac{1}{2}$ for $t \in [T, T + 1]$ and $c > c_0$.*

Proof. In (V. S. Borkar and S. P. Meyn 2000, Lemma 4.4) the statement is shown for every $\varepsilon > 0$, i.e. with ε in place of $\frac{1}{2}$. □

We are now ready to prove the distributed BMT. As mentioned before, the line of argument differs from the traditional BMT since the scaling sequence $s(m)$ does not scale the initial points of every T-length interval to the unit sphere. In the traditional BMT setting, one shows that $\frac{\|x(T_{m+1})\|}{\|x(T_m)\|} < \varepsilon \in (0, 1)$ for m sufficiently large. Using the same reasoning, here we only obtain

$$\frac{\|x(T_{m+1})\|}{\|x(T_m)\|} = \lim_{t \nearrow T_{m+1}} \|\hat{x}(t)\| \frac{s(m)}{\|x(T_m)\|} < \varepsilon \frac{s(m)}{\|x(T_m)\|}, \tag{2.40}$$

for m sufficiently large. Technically, $\frac{s(m)}{\|x(T_m)\|}$ can be arbitrarily large, as $\|x(T_m)\|$ may we small, while $s(m)$ is unbounded when assuming instability. Hence, one *can not choose* ε, such that the right-hand side (2.40) is less than 1, whenever $\|\overline{x}(T_m)\| > c_0$ with c_0 from Lemma 2.11. In contrast, $\frac{s(m)}{\|x(T_m)\|} = 1$ for the traditional BMT, and one concludes that $\|\overline{x}(T_m)\|$ falls back at an exponential rate to the ball of radius c_0, which leads to a contradiction with the assumed instability. This line of argument is not applicable here, and we have developed a fundamentally new one to prove Theorem 2.1.

Proof of Theorem 2.1. Fix $T > 0$ from Lemma 2.11 and for this T create the approximate T-length intervals $[T_m, T_{m+1}]$ for all $m \geq 0$ as described in (2.10). Then fix a sample point where Lemma 2.6 and Lemma 2.10 hold. Recall the scaling sequence $s(m) = \max\{\sup_{l \leq m} \|\overline{x}(T_l)\|, 1\}$ and suppose that $\sup_{m \geq 0} \|\overline{x}(T_m)\| < \infty$ does not hold. Then, by construction, $s(m) \nearrow \infty$ monotonically.

We will now consider those time intervals where the scaling sequence equals the norm of the trajectory at the beginning of the T-length intervals, i.e., those m where $s(m) = \|\overline{x}(T_m)\|$. That is, we consider those time steps where $\|\hat{x}(T_m)\| = 1$. This yields a subsequence of interval indices $\{\tilde{m}(k)\}_{k \geq 0} \subset \{m\}_{m \geq 0}$, such that $\|\hat{x}(T_{\tilde{m}(k)})\| = 1$ for all $k \geq 0$, and $s(l) = \overline{x}(T_{\tilde{m}(k)})$ for all $l \in \{\tilde{m}(k), \ldots, \tilde{m}(k + 1) - 1\}$.

Recall now that $x^m(t)$, $t \in [T_m, T_{m+1}]$, are the unique solutions to $\dot{x}(t) = h_{s(m)}(x(t))$ with initial condition $x^m(T_m) = \hat{x}(T_m) = \frac{\overline{x}(T_m)}{s(m)}$. Since $s(m) \nearrow \infty$, there is some k', such that $s(\tilde{m}(k)) > c_0$ with c_0 from Lemma 2.11 for all $k \geq k'$. *Most importantly, for any $k \geq k'$ we now consider the last interval, $\tilde{m}(k+1)-1$, from the set of intervals $\{\tilde{m}(k), \ldots, \tilde{m}(k+1)-1\}$ where $s(\tilde{m}(k))$ is used as the scaling factor.* By the monotone scaling sequence, $x^{\tilde{m}(k+1)-1}(T_{\tilde{m}(k+1)-1})$ is on the closed unit ball and thus Lemma 2.11 shows that $\|x^{\tilde{m}(k+1)-1}(T_{\tilde{m}(k+1)})\| < \frac{1}{2}$. Further, Lemma 2.10 shows that there is some k'', such that

$$\sup_{t \in [T_{\tilde{m}(k+1)-1}, T_{\tilde{m}(k+1)}]} \|\hat{x}(t) - x^{\tilde{m}(k+1)-1}(t)\| < \frac{1}{2}, \quad k \geq k''. \tag{2.41}$$

The ratio of successive scaling factors, therefore, satisfies

$$\begin{aligned} \frac{s(\tilde{m}(k+1))}{s(\tilde{m}(k))} = \frac{\overline{x}(T_{\tilde{m}(k+1)})}{\overline{x}(T_{\tilde{m}(k)})} &= \lim_{t \nearrow T_{\tilde{m}(k+1)}} \|\hat{x}(t)\| && (2.42) \\ &\leq \lim_{t \nearrow T_{\tilde{m}(k+1)}} \|x^{\tilde{m}(k+1)-1}(t)\| + \lim_{t \nearrow T_{\tilde{m}(k+1)}} \|\hat{x}(t) - x^{\tilde{m}(k+1)-1}(t)\| && (2.43) \\ &< \frac{1}{2} + \frac{1}{2} = 1 && (2.44) \end{aligned}$$

for all $k > \max(k', k'')$, i.e., $s(\tilde{m}(k+1)) < s(\tilde{m}(k))$. This is the required contradiction since $s(m)$ was constructed as monotonically increasing. Hence, $\sup_{m \geq 0} s(m) < \infty$ almost surely and the theorem follows from Lemma 2.8. □

The crucial point of the new line of argument for Theorem 2.1 is that it completely removes the use of Gronwalls inequality. We derive the required contradiction by "simply" showing that the adaptive monotone scaling sequence would have to eventually decrease if x_n is unstable.

For the remainder of this section, we now move beyond the results presented in (Redder, Ramaswamy, and Karl 2023) and discuss some interesting observations and extensions based on the proof of Theorem 2.1. First, observe that to obtain the contradiction for Theorem 2.1, it is sufficient that the limes superior of the right-hand side in (2.43) is strictly less than one. It appears that one can use this observation to conclude that the BMT holds, provided merely that the origin is stable and not necessarily asymptotically stable. We discuss this in the following remark

Remark 2.2.3. *Observe that a weaker version of Lemma 2.11 is sufficient for the conclusion in Theorem 2.1. It is enough that there exist $c_0 \geq 1$ and $T > 0$ and a sequence $\varepsilon(c) > 0$, such that for all initial conditions x on the closed unit ball,*

$$\|\phi_c(x, t)\| < 1 + \varepsilon(c), \quad t \in [T, T+1] \tag{2.45}$$

for and $c > c_0$, with $\varepsilon(c) \to 0$ as $c \to \infty$. With Lemma 2.10 and (2.43) it then follows that

$$\limsup_{k \to \infty} \frac{s(\tilde{m}(k+1))}{s(\tilde{m}(k))} < 1, \tag{2.46}$$

contradicting monotonicity, which would imply that $\liminf_{k\to\infty} \frac{s(\tilde{m}(k+1))}{s(\tilde{m}(k))} \geq 1$*, and thus implying that* $\sup_{m\geq 0} s(m) < \infty$ *almost surely.*

From the proof of Lemma 2.11, one can now see that to verify (2.45) it is sufficient that there is arbitrary neighborhood $\mathcal{U}$ *of the origin and a* $T > 0$*, such that every solution of* $\dot{x}(t) = h_\infty(x(t))$ *with initialization on* $\mathcal{U}$ *stays inside the open unit Ball for* $t > T$*. This is clearly guaranteed if the origin is stable, i.e., if, for some initial value close to the origin, a solution will eventually stay close to the origin. In other words, it is not necessary that the origin is locally asymptotically stable. With this generalization, the BMT now includes drifts of the form*

$$h(x) = \begin{pmatrix} x^2 \\ -x^1 \end{pmatrix} \tag{2.47}$$

with $x = (x^1, x^2)$*. The ODE (2.47) is the well-known example of an ODE that is stable, but not asymptotically stable (Bhatia and Szegö 2006), where trajectories are circles centered at the origin.*

Remark 2.2.3 sketches that asymptotic stability in Assumption 2.1.2 can be replaced merely by stability. Importantly, this new version includes sublinear drifts where $h(x)$ is hölder continuous, i.e. $\|h(x) - h(y)\| \leq L\|x - y\|^\alpha$ for some $\alpha \in [0, 1)$. In this case, $h_\infty(x) = 0$ for all $x \in \mathbb{R}^d$ and $\dot{x}(t) = 0$ is clearly not asymptotically stable to the origin but merely stable.

We will now discuss some further extensions while sticking to the asymptotic stability setup.

2.2.4 Further extensions

A weaker version of Assumption 2.1.2

The new line of argument in Theorem 2.1 reveals how to further weaken assumption of the original BMT: the original BMT requires that $\frac{h(cx)}{c}$ converges pointwise to some limit $h_\infty(x)$ as $c \to \infty$, where h_∞ is asymptotically stable to the origin. We show that it is merely required that a scaling sequence $c_n \nearrow \infty$ exists such that $\lim_{n\to\infty} \frac{h(c_n x)}{c_n}$ is asymptotically stable to the origin (Theorem 2.12). Previously, the limit needed to exist for any scaling sequence drifting to infinity.

We can now propose a weaker version of Assumption 2.1.2 that also applies to the traditional BMT setting.

Assumption 2.2.1. *The functions* $h_c(x) := \frac{h(cx)}{c}, c \geq 1$*,* $x \in \mathbb{R}^d$*, satisfy that there exists a sequence* $c_n \nearrow \infty$*, such that* $h_{c_n}(x) \to h_\infty(x)$ *compactly as* $n \to \infty$ *for some* $h_\infty \in C(\mathbb{R}^d)$*, where the ODE*

$$\dot{x}(t) = h_\infty(x(t)) \tag{2.48}$$

has the origin as a asymptotically stable equilibrium.

Notice that by the Lipschitz condition of h, a sequence c_n always exists such that $h_{c_n}(x)$ converges compactly. Assumption 2.1.2, on the other hand, requires that any scaling sequence approaches a suitable limiting function. The essence of the generalization is that we use a monotone scaling sequence designed using c_n.

Theorem 2.12. *Consider Theorem 2.1, with Assumption 2.2.1 instead of Assumption 2.1.2, then* $\sup_{n\geq 0}\|x_n\| < \infty$ *a.s..*

We conjecture that the ideas behind Assumption 2.2.1 can also be applied to other generalizations of the BMT, e.g., for set-valued recursive inclusions.

Remark 2.2.4. *In their BMT paper, V. S. Borkar and S. P. Meyn (2000) mention that when $h_c(x)$ does not converge pointwise, then one can omit the fluid model as long as one can show the statement of Lemma 2.11. In Remark 2.2.3, we saw that the essence for stability is (2.45).*

Further, in the proof of Theorem 2.12, one sees how one can keep the fluid model by working with a subsequence c_n. Since a convergent subsequence can always be extracted, Corollary 2.13 below clarifies that the convergence of h_c is immaterial for the stability of h; it only matters that h_c is "eventually" attracting towards the origin.

Remark 2.2.5. *A generalization of the traditional BMT was proposed by Ramaswamy and Shalabh Bhatnagar (2017) using a version of the BMT for set-valued recursive inclusions. For every $x \in \mathbb{R}^d$, define $h_\infty(x) := \overline{co}\{\limsup_{c\to\infty}\{h_c(x)\}\}$, where $\overline{co}$ denotes the convex closure. The stability condition given by Ramaswamy and Shalabh Bhatnagar (2017) is that the differential inclusion $\dot{x}(t) \in h_\infty(x(t))$ has an attracting set. While Assumption 2.1.2 implies both this condition and Assumption 2.2.1, it appears that neither of the two implies the other; Assumption 2.2.1 relies on one single convergent subsequence, while the set-valued condition relies on all convergent subsequences.*

We can now give a simple stability condition based on Theorem 2.12, which we apply in Chapter 3. Notice that by Rademacher's theorem, $h(x)$ is differentiable almost everywhere. Thus, the Jacobian matrix $Dh(x)$ exists for almost every $x \in \mathbb{R}^d$. Denote the largest eigenvalue of a matrix by $\lambda_{\max}(\cdot)$.

Corollary 2.13. *Suppose that Assumption 2.1.1, 2.1.3-2.1.5 hold and in addition that*

$$\limsup_{\|x\|\to\infty} \lambda_{\max}(Dh(x)) < 0, \tag{2.49}$$

then, $\sup_{n\geq 0}\|x_n\| < \infty$ *a.s.*

With the insights form Remark 2.2.3, (2.49) can be weakened "$\leq$" instead of "$<$".

A weaker version of Assumption 2.1.3

We simplified the presentation of the previous sections using Assumption 2.1.3. A more general version that is also useful is to replace Assumption 2.1.3 by

$$\mathbb{E}\left[\|M_{n+1}^i\|^2 \mid \mathcal{F}_n\right] \leq K^2\left(1+\sup_{0\leq k_{ij}\leq\tau_{ij}(n)}\|(x_{n-k_{i1}}^1,\ldots,x_{n-k_{iD}}^D)\|^2\right) \text{ for some } K>0. \tag{2.50}$$

The presented analysis of distributed SA will hold with minor modifications with (2.50) in place of the second part of Assumption 2.1.3. Notably, (2.25) will be replaced by

$$\mathbb{E}\left[\|\hat{M}_{n+1}^i\|^2\right]^{\frac{1}{2}} \leq K\left(1+\mathbb{E}\left[\|\hat{x}(t(n)\|^2\right]^{\frac{1}{2}}+\sum_{j=1}^{D}\mathbb{E}\left[\sup_{0\leq k_{ij}\leq\tau_{ij}(n)}\left(\frac{\|x_n^j-x_{n-k_{ij}}^j\|}{s(m(n))}\right)^2\right]^{\frac{1}{2}}\right) \tag{2.51}$$

and the recursive inequality in L_2 will arise for $\sum_{i,j=1}^{D}\mathbb{E}\left[\sup_{0\leq k_{ij}\leq\tau_{ij}(n)}\left(\frac{\|x_n^j-x_{n-k_{ij}}^j\|}{s(m(n))}\right)^2\right]^{\frac{1}{2}}$.

2.3 Stability of stochastic approximations with momentum

This section shows how the tools and techniques used to prove the distributed BMT can be applied to SA with momentum. For easy reference, we recall the SA iteration with Polyak's heavy ball momentum from (1.13):

$$\begin{aligned} x_{n+1} &= x_n + a(n)m_k \\ m_k &= \beta m_{k-1} + (1-\beta)g(x_k) \end{aligned} \tag{2.52}$$

where $m_{-1}=0$, momentum parameter $\beta\in[0,1)$ and $g(x_k) \coloneqq h(x_k)+M_{k+1}$ with drift h and Martingale noise M_{k+1} as before.

Theorem 2.14 (BMT for SA with heavy-ball momentum)**.** *Under Assumption 2.1.1-2.1.3 for the stochastic heavy-ball iteration (2.52) (i.e., with $\tau_{ij}(n)=0$ for all $n\geq 0$) and $a(n)\in\mathcal{O}(n^q)$, $q\in(1/2,1]$, it follows that (2.52) is stable almost surely.*

Corollary 2.15. *Under the assumptions of Theorem 2.14, the stochastic heavy-ball iteration (2.52) converges almost surely to a potential sample path-dependent compact connected internally chain transitive invariant set of the ODE (2.3).*

As mentioned in the introduction, the proof idea for Theorem 2.14 is to rewrite the moving average of the past drift terms using a determinism AoI sequence:

$$x_{n+1} = x_n + a(n)(1-\beta)\left[\sum_{i=n-\tau(n)}^{n}\beta^{n-i}g(x_i)\right] + a(n)(1-\beta)\left[\sum_{i=0}^{n-\tau(n)-1}\beta^{n-i}g(x_i)\right], \tag{2.53}$$

with deterministic AoI sequence $\tau(n) \coloneqq \lceil \frac{n}{\sum_{k=0}^{n} a(n)} \rceil$. Further, define $c(n) \coloneqq \sum_{i=n-\tau(n)}^{n} \beta^{n-i}$ and define the tail error $\delta_n \coloneqq \frac{1}{c(n)} \sum_{i=1}^{n-\tau(n)-1} \beta^{n-i} g(x_i)$. It is now not difficult to show that

$$\sup_{n\geq 0} c(n) = \lim_{n\to\infty} c(n) = \frac{1}{1-\beta} \text{ and } \sup_{n\geq 0} \frac{1}{c(n)} \leq 1. \tag{2.54}$$

The first important step is to show that the tail error vanishes almost surely.

Lemma 2.16. $\|\delta_n\| \in o(1)$.

Proof sketch. For $a(n) \in \mathcal{O}(n^q)$, $q \in (1/2, 1]$), Gronwalls inequality (see A.1) shows that $\mathbb{E}\left[\|x_n\|^2\right]^{\frac{1}{2}} \in \mathcal{O}\left(\exp\left(n^{1-p}\right)\right)$. Further, by construction $\tau(n) \geq cn^q$ for some $c > 0$. It then follows that

$$\mathbb{E}\left[\|\delta_n\|^2\right]^{\frac{1}{2}} \in \mathcal{O}\left(\sum_{k=0}^{n-cn^q} \beta^{n-k} \exp\left(k^{1-q}\right)\right) \tag{2.55}$$

$$\in \mathcal{O}\left(\beta^{cn^q} \sqrt{n} \exp\left(\sqrt{n}\right)\right) \in o(1), \tag{2.56}$$

where the last step uses that $q > \frac{1}{2}$. In the same way, using the martingale convergence theorem, we can show that $\|\delta_n\| \in o(1)$. □

Next, write (2.53) as

$$x_{n+1} = x_n + \tilde{a}(n)\left[h(x_n) + \tilde{M}_{n+1} + e_n + \delta_n\right]. \tag{2.57}$$

with $\tilde{a}(n) \coloneqq a(n)(1-\beta)c(n)$ and

$$e_n \coloneqq \frac{1}{c(n)} \sum_{i=n-\tau(n)}^{n-1} \beta^{n-i}\left(h(x_i) - h(x_n)\right), \quad \tilde{M}_{n+1} \coloneqq \frac{1}{c(n)}\left(\sum_{i=n-\tau(n)}^{n} \beta^{n-i} M_{i+1}\right). \tag{2.58}$$

Notice the similarity to Equation (2.4) with the addition of the error δ_n. Equation (2.57) can now be analyzed with the tools and techniques presented in Section 2.2. Specifically, (2.57) is a SA iteration with martingale difference noise $\tilde{M}_{n+1}$, an error e_n due to the deterministic AoI sequence and the additional error $\delta_n \in o(1)$. Theorem 2.14 is now almost an immediate consequence of the steps used to prove Theorem 2.1. However, small changes are necessary, outlined in the appendix below.

2.4 Discussion and related work

Theorem 2.1 and Theorem 2.14 fill gaps in the existing literature, which are: 1) A stability theorem for distributed stochastic approximations with unbounded drift and unbounded stochastic AoI 2) A convergence theorem for distributed stochastic approximations where information delays have only an arbitrary moment bound 3) Stability theorem for general stochastic approximation

iterations with heavy-ball momentum and a fixed momentum parameter. In particular, the core assumption that makes the distributed BMT possible is Assumption 2.1.5. Further, point 2) will be completed based on the properties of AoI and the existence of AoI-dominated random variables with some moment bound established in Part II how to verify Assumption 2.1.5. Finally, we believe that through the ideas surrounding Remark 2.2.3, we identified a significant potential to weaken the traditional local asymptotic stability assumption in the BMT to merely local stability. We will now review and discuss related work on distributed SA and SA with momentum.

If not otherwise stated, all of the following works consider decaying stepsizes that are not summable but square summable. One of the earliest works on asynchronous distributed SA dates back to D. Bertsekas (1982), who presented an abstract dynamic programming approach in the presence of bounded communication delays. John N Tsitsiklis (1994) then proposed the first asynchronous distributed SA algorithm for potentially unbounded delay, with assumptions tailored towards Q-learning. The first asynchronous distributed SA of the form (1.5), which from today's point of view is in the standard form of an SA iteration, was considered by V. S. Borkar (1998). Here, almost sure convergence is shown *assuming stability* and that the delays satisfy, for some $p > 0$, $\sup_{n\geq 0} \mathbb{E}\left[\tau_{ij}^p(n) \mid \tau_{ij}(k), k \leq n-1\right] < \infty$ a.s. This assumption is quite restrictive compared to the unconditional version assumed herein, as the following example shows.

Example 2.4.1. *Consider the fundamental AoI process* (1.8) *introduced in Chapter 1 and suppose the events $A(n)$ are i.i.d. Bernoulli*(1/2)*. Then for any $p > 0$,*

$$\sup_{n\geq 0} \mathbb{E}\left[\tau^p(n) \mid \tau(k), k \leq n-1\right] = \frac{1}{2} + \frac{1}{2}\sup_{n\geq 0} \tau^p(n-1) = \infty \tag{2.59}$$

The last inequality follows since, for every sample path, there is a non-zero probability that $\tau(n)$ reaches any desired level.

Example 2.4.1 shows that V. S. Borkar (1998) condition practically requires a delay moment bound independent of the delay at the previous time step. Up until the works presented herein, the literature then required $\sup_{n\geq 0} \mathbb{E}\left[\tau_{ij}^p(n)\right] < \infty$ for some $p > 1$ for almost sure convergence to an equilibrium while assuming stability (V. Borkar 2022, Chapter 6).

As mentioned before, Shalabh Bhatnagar (2011) gave a stability theorem for general asynchronous SAs with bounded delays. Specifically, a version of the BMT was shown to hold under bounded delays and slightly stronger martingale noise assumptions than typical. We do not require any of those restrictions. The key insights that enable this progress are the disclosure of the crucial interplay between the algorithm AoI and algorithm stepsize and the recursive structure of the SA drift errors caused by AoI. But, it should be noted that the analysis presented by Shalabh Bhatnagar (2011), or the more recent version by Yu, Wan, and R. S. Sutton 2023, also consider asynchronous updates due to clocks with different speeds. This can be included in the

framework under the assumptions presented therein. Finally, in (Ramaswamy, Shalabh Bhatnagar, and Daniel E Quevedo 2020), a stability and convergence theory for asynchronous SA with biases was developed that also incorporates delays when the $p > 1$ moments are bounded. The crucial stability condition is that the distance to an associated projective scheme is bounded almost surely, which holds for approximate value iteration without delays. However, this condition does not naturally hold for asynchronous gradient-based optimization or momentum-SA methods studied herein. Further, the boundedness condition has to be verified, considering delays, which will practically require bounded delays, bounded drift or another condition to be applicable.

Beyond general stochastic approximation iterations, several works have been on distributed gradient-based methods with delays. We review the relevant literature in the next chapter, which focuses on distributed SGD. We will now close this chapter with related work on SA with momentum. Momentum methods have been extensively studied for gradient-based schemes since the seminal works by Polyak (1964) and Y. E. Nesterov (1983). During the last decade, momentum-based methods have received attention due to their success in machine learning (Sutskever et al. 2013; Wilson et al. 2017). Notably, momentum has been shown to improve the rate of convergence and has also been shown to help in the avoidance of saddle points (Jin et al. 2017).

One of the most studied momentum schemes is SGD with Polyaks heavy-ball momentum, known as the stochastic heavy ball (SHB) method. SHB was studied extensively in the last years (Gadat, Panloup, and Saadane 2018; Y. Liu, Gao, and W. Yin 2020; Sebbouh, Gower, and Defazio 2021; Jun Liu and Yuan 2022). Notably, Barakat and P. Bianchi (2021) recently presented a detailed analysis of momentum-based gradient schemes using a dynamical systems approach. Jun Liu and Yuan (2022) proved for the first time that SHB's last iterate converges almost surely to a stationary point of non-convex objective functions. The assumptions are typical for a stochastic approximation analysis (V. Borkar 2022, Sec. 2). The natural question is, therefore, whether general stochastic approximations with heavy ball momentum converge to an equilibrium. We gave an answer to this question with Corollary 2.15, which establishes the convergence of the scheme to an invariant set under the BMT condition. In other words, when the drift function of the SA momentum scheme has only isolated equilibria and the conditions of Corollary 2.15 hold, then (1.13) converges to an equilibrium.

Stochastic approximation iterations with momentum have also recently seen more attention due to their use in reinforcement learning. Devraj, Bušić, and S. Meyn (2019) presented a matrix momentum SA method with optimal asymptotic covariance for a class of linear stochastic approximations. Mou et al. (2020) studied Polyak-Ruppert averages for linear SA with heavy-ball momentum. A specific one-dimensional SA estimator with heavy-ball momentum was studied for estimating change rates of web pages by Avrachenkov, Patil, and Thoppe (2022). Finally,

Deb and Shalabh Bhatnagar (2022) and Deb and Shalabh Bhatnagar (2021) presented a BMT style theorem for multi-time scale stochastic approximations. The authors then analyze linear temporal difference learning with heavy-ball momentum, which can be viewed as a multi-time scale stochastic approximation. Notably, the above analyses of linear SA with heavy-ball momentum use time-varying momentum parameter $\beta_n \nearrow 1$. Instead, we are not restricted to linear SA and allow an arbitrary fixed momentum parameter. Furthermore, the analysis herein can also be extended to momentum parameters $\beta_n \nearrow 1$, which will be essential to study with moving average form a version of Nesterov acceleration applied to SA Su, Boyd, and Candes 2016.

2.5 Proofs of Chapter 2

Proof of Lemma 2.3. Define $B := \sup_{n\geq 0} b_n$. Let $n \geq 0$, then for all $m \leq n$, $y_m \leq c_n B + L\sum_{k=0}^{m-1} a_k y_k$, as c_n is monotonically increasing. The discrete Gronwall inequality thus implies that

$$y_n \leq c_n B e^{Lt(n)}. \tag{2.60}$$

Now consider $N < \infty$ as defined by condition (2.18) in Lemma 2.3. Then,

$$\begin{aligned} y_{N+1} &\leq c_{N+1}b_{N+1} + L\sum_{k=N+1-\Delta_{N+1}}^{N} a_k y_k && (2.61)\\ &\leq c_{N+1}b_{N+1} + c_N B e^{Lt(N)} L\sum_{k=N+1-\Delta_{N+1}}^{N} a_k && (2.62)\\ &\leq c_{N+1}b_{N+1} + c_N B(e^{Lt(N)} - 1) && (2.63)\\ &\leq c_{N+1} B e^{Lt(N)}, && (2.64) \end{aligned}$$

where the second inequality uses (2.60) and that both c_n and $t(n)$ are increasing; the third inequality applies the definition of N; and the last inequality again uses that c_n is increasing and that $b_{N+1} \leq B$. It now follows by induction that

$$y_n \leq c_n B e^{Lt(N)} \tag{2.65}$$

for all $n \geq 0$. By using this inequality in (2.17), we obtain

$$\begin{aligned} y_n &\leq b_n c_n + L\sum_{k=n-\Delta_k}^{n-1} a_k y_k, && (2.66)\\ &\leq b_n c_n + L\sum_{k=n-\Delta_k}^{n-1} a_k c_k B e^{Lt(N)} && (2.67)\\ &\leq c_n\left(b_n + BLe^{Lt(N)}\left(\sum_{k=n-\Delta_n}^{n-1} a_k\right)\right). && (2.68) \end{aligned}$$

□

Proof of Lemma 2.4. Equation (2.35) leads to

$$\sum_{i,j=1}^{D} \mathbb{E}_z \left[\left(\frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))} \right)^2 \right]^{\frac{1}{2}} \leq 2KD^2 \left(\sum_{k=n-\Delta_z(n)}^{n-1} a(k) \right) \left(1 + \sup_{k \leq n-1} \mathbb{E}_z \left[\|\hat{x}(t(k))\|^2 \right]^{\frac{1}{2}} \right) + 2KD \sum_{k=n-\Delta_z(n)}^{n-1} a(k) \sum_{i,j=1}^{D} \mathbb{E}_z \left[\left(\frac{\|x_k^j - x_{k-\tau_{ij}(k)}^j\|}{s(m(n))} \right)^2 \right]^{\frac{1}{2}}. \tag{2.69}$$

Define $y_n \coloneqq \sum_{i,j=1}^{D} \mathbb{E}_z \left[\left(\frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))} \right)^2 \right]^{\frac{1}{2}}$, $c_n \coloneqq \left(1 + \sup_{k \leq n-1} \mathbb{E}_z \left[\|\hat{x}(t(k))\|^2 \right]^{\frac{1}{2}} \right)$, $L \coloneqq 2KD$, $b_n \coloneqq 2KD^2 \left(\sum_{k=n-\Delta_z(n)}^{n-1} a(k) \right)$ as well as $a_n \coloneqq a(n)$ and $\Delta_n \coloneqq \Delta_z(n)$ as evident. The lemma now follows from Lemma 2.3. □

Recall that we used $m(n)$ to map from a discrete time-step n to the corresponding interval $[T_{m(n)}, T_{m(n)+1})$ in continuous time. Hence, $m(n)$ is clearly not one-to-one. However, having an "inverse" of $m(n)$ will now become useful, i.e., a map $n(m)$ that sends the m-th interval to the discrete time-step n corresponding to T_m. Specifically, $T_m = t(n(m)) = \sum_{i=0}^{n(m)-1} a(i)$ for a unique strictly increasing sequence $n(m) \nearrow \infty$.

Proof of Lemma 2.5. Fix $m \geq 0$ and $n(m) \leq n < n(m+1)$. Insert Lemma 2.4 into (2.28) for $\mathbb{E}_z[\cdot]$, then

$$\mathbb{E}_z \left[\|\hat{x}(t(n+1))\|^2 \right]^{\frac{1}{2}} \leq \mathbb{E}_z \left[\|\hat{x}(t(n))\|^2 \right]^{\frac{1}{2}} (1 + a(n)K_1) + a(n) \left(K_1 + K_2K_3 \left(1 + \sup_{k \leq n-1} \mathbb{E}_z \left[\|\hat{x}(t(k))\|^2 \right]^{\frac{1}{2}} \right) \left(\sum_{k=n-\Delta_z(n)}^{n-1} a(k) \right) \right). \tag{2.70}$$

Since $\sum_{n=n(m)}^{n(m+1)-1} a(n) \leq T+1$ and $\|\hat{x}(t(n(m)))\| \leq 1$, a simple recursion shows that

$$\mathbb{E}_z \left[\|\hat{x}(t(n+1))\|^2 \right]^{\frac{1}{2}} \leq \exp(K_1(T+1))(1 + K_1(T+1)) + \exp(K_1(T+1))K_2K_3 \left(1 + \sup_{k \leq n-1} \mathbb{E}_z \left[\|\hat{x}(t(k))\|^2 \right]^{\frac{1}{2}} \right) \left(\sum_{k=n(m)}^{n} a(k) \left(\sum_{l=k-\Delta_z(k)}^{k-1} a(l) \right) \right), \tag{2.71}$$

where we used that $1 + a(n)K_1 \leq \exp(a(n)K_1)$. As $\mathbb{E}_z \left[\|\hat{x}(t(0))\|^2 \right]^{\frac{1}{2}} < \infty$, it follows from (2.71) that $\mathbb{E}_z \left[\|\hat{x}(t(n))\|^2 \right]^{\frac{1}{2}} < \infty$ for every fixed $n \geq 0$. Further, it follows from Assumption 2.1.5 and $\sum_{n=n(m)}^{n(m+1)-1} a(n) \leq T+1$ that $\sum_{k=n(m)}^{n} a(k) \left(\sum_{l=k-\Delta_z(k)}^{k-1} a(l) \right) \in o(1)$. The lemma statement is

now immediate from these two conclusions, (2.71) and the following simple property that can be shown by induction: Let $\{z_n\}, \{b_n\}$ be a non-negative sequences in $\mathbb{R}^d$, such that $z_{n+1} \leq C(1 + b_n \sup_{k \leq n} z_k)$ with $b_n \in o(1)$, then $\sup_{n \geq 0} z_n < \infty$.

□

Proof of Lemma 2.6. By Remark 2.2.1, it is enough to show that $\mathbb{P}\left(\hat{\zeta}_n \text{ converges } \mid E_z\right) = 1$ for all $z \geq 0$. By the convergence theorem for square-integrable martingales (see A.2) it is enough to show that $\sum_{n \geq 0} \mathbb{E}_z \left[\|a(n)\hat{M}_{n+1}\|^2 \mid \mathcal{F}_n\right] < \infty$ a.s. for the filtration $\mathcal{F}_n$ defined in Assumption 2.1.3. By contradiction, it is enough to show that $\mathbb{E}_z \left[\sum_{n \geq 0} a(n)^2 \mathbb{E}_z \left[\|\hat{M}_{n+1}\|^2 \mid \mathcal{F}_n\right]\right] < \infty$. Using (2.25) for $\mathbb{E}_z[\cdot]$, we have that

$$\mathbb{E}_z \left[\sum_{n \geq 0} a(n)^2 \mathbb{E}_z \left[\|\hat{M}_{n+1}\|^2 \mid \mathcal{F}_n\right]\right] = \sum_{n \geq 0} a(n)^2 \mathbb{E}_z \left[\|\hat{M}_{n+1}\|^2\right] \tag{2.72}$$

$$\leq \sum_{n \geq 0} a(n)^2 \frac{K^2}{\mathbb{P}(E_z)} \left(D\left(1 + \mathbb{E}_z \left[\|\hat{x}(t(n))^2\|\right]^{\frac{1}{2}}\right) + \sum_{i,j=1}^{D} \mathbb{E}_z \left[\left(\frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))}\right)^2\right]^{\frac{1}{2}} \right)^2, \tag{2.73}$$

Lemma 2.4, Lemma 2.5 and the square summability of $a(n)$ (Assumption 2.1.4) therefore imply that (2.73) is finite and the statement follows. Notice that the equality follows by the tower property, since $E_z \in \mathcal{F}_n \cap E_z$ as $\mathcal{F}_n$ is a σ-algebra for all $n \geq 0$. □

Proof of Corollary 2.7. We have that

$$\frac{1}{s(m(n))} \sum_{k=0}^{n} a(k) M_{k+1}^j = \frac{1}{s(m(n))} \sum_{k=0}^{n} s(m(k)) a(k) \hat{M}_{k+1}^j. \tag{2.74}$$

Now for every sample point, there are two scenarios:

1) $s(m(n))$ is bounded, then $\sum_{k=0}^{n} s(m(k))a(k)\hat{M}_{k+1}^j$ converges by Abel's test for infinite series.
2) $s(m(n))$ is unbounded, then $\frac{1}{s(m(n))} \sum_{k=0}^{n} s(m(k))a(k)\hat{M}_{k+1}^j \to 0$ by Kronecker's lemma for infinite series.

It follows that $\frac{1}{s(m(n))} \sum_{k=0}^{n} a(k) M_{k+1}^j$ converges almost surely. As all $n - \tau_{ij}(n) \to \infty$ a.s. implied by Assumption 2.1.5, the corollary follows. □

Proof of Lemma 2.8. Fix an interval $m > 0$ and $n(m) \leq n < n(m+1)$. Then the rescaled iteration satisfies

$$\hat{x}(t(n+1)) = \hat{x}(t(n)) + a(n) \left[h_{s(m)}(\hat{x}(t(n))) + \hat{e}_n + \hat{M}_{n+1}\right], \tag{2.75}$$

using the functions $h_c(\cdot)$ from Assumption 2.1.2. Moreover,

$$\|\hat{e}_n\| \leq L \sum_{i,j=1}^{D} \frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m)}. \tag{2.76}$$

To simplify the presentation, define a point-wise upper bound for all $\tau_{ij}(n)$, i.e., define $\tau(n) \coloneqq \max\{\tau_{ij}(n) \mid 1 \leq i,j \leq D\}$. Notice that $\tau(n)$ is a stochastic bound in contrast to the deterministic bounds used in Lemma 2.5. As the number of iterations is finite ($D < \infty$), it follows from Assumption 2.1.5 that $\sum_{k=n-\tau(n)}^{n-1} a(k) \to 0$ almost surely. Starting from (2.9), we now obtain that

$$\begin{aligned}\sum_{i,j=1}^{D} \frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))} &\leq KD^2 \left(\sum_{k=n-\tau(n)}^{n-1} a(k) \right) (1 + \sup_{k \leq n-1} \|\hat{x}(t(k))\|) \\ &+ \sum_{i,j=1}^{D} \| \frac{1}{s(m(n))} \sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) M_{k+1}^j \|, \\ &+ KD \sum_{k=n-\tau(n)}^{n-1} a(k) \sum_{i,j=1}^{D} \frac{\|x_k^j - x_{k-\tau_{ij}(k)}^j\|}{s(m(k))},\end{aligned} \tag{2.77}$$

where we again used the monotonicity of $s(m(n))$. Please notice the similarity to (2.35) and the recursive structure arising in (2.77). Also, notice that the third term in (2.77) converges to zero by Corollary 2.7. As before we use Lemma 2.3 to conclude that there is a constant $K_4 > 0$, such that

$$\sum_{i,j=1}^{D} \frac{\|x_n^j - x_{n-\tau_{ij}(n)}^j\|}{s(m(n))} \leq K_4 \left(1 + \sup_{k \leq n-1} \|\hat{x}(t(k))\| \right) \left(\sum_{k=n-\tau(n)}^{n-1} a(k) \right) \tag{2.78}$$

Iterating the rescaled iteration (2.75) now yields, for $0 < k \leq n(m+1) - n(m)$,

$$\begin{aligned}\hat{x}(t(n(m)+k)) &= \hat{x}(t(n(m))) + \sum_{i=0}^{k-1} a(n(m)+i) h_{s(m)}(\hat{x}(t(n(m)+i))) \\ &+ \sum_{i=0}^{k-1} a(n(m)+i) \hat{e}_{n(m)+i} + \left(\hat{\zeta}_{n(m)+k} - \hat{\zeta}_{n(m)} \right),\end{aligned} \tag{2.79}$$

with $\|h_{s(m)}(\hat{x}(t(n(m)+i)))\| \leq K\left(1 + \|\hat{x}(t(n(m)+i))\|\right)$. The convergence of the accumulated rescaled martingale noise (Lemma 2.6) now implies that $\sup_{n \geq 0} \|\hat{\zeta}_n\| < \infty$ a.s.. Hence, there is sample path dependent constant $B > 0$ such that $\sup_{n \geq 0} \|\hat{\zeta}_n\| \leq B$. With

$$\sum_{0 \leq i < n(m+1) - n(m)} a(n(m)+i) \leq T + 1$$

and $\|\hat{x}(t(n(m)))\| \leq 1$, it follows that

$$\begin{aligned}\|\hat{x}(t(n(m)+k))\| &\leq 1 + K(T+1) + K \sum_{i=0}^{k-1} a(n(m)+i) \|\hat{x}(t(n(m)+i))\| \\ &+ \sum_{i=0}^{k-1} a(n(m)+i) \|\hat{e}_{n(m)+i}\| + 2B\end{aligned} \tag{2.80}$$

The traditional discrete Gronwall inequality (see A.1) now shows that

$$\|\hat{x}(t(n(m)+k))\| \leq \left(1 + K(T+1) + 2B + \sum_{i=0}^{k-1} a(n(m)+i)\|\hat{e}_{n(m)+i}\|\right) \exp(K(T+1)) \quad (2.81)$$

for $0 < k \leq n(m+1) - n(m)$. By combining (2.76), (2.78) and (2.81) we arrive at

$$\begin{aligned}\|\hat{x}(t(n(m)+k))\| &\leq (1 + K(T+1) + 2B)\exp(K(T+1)) \\ &+ \exp(K(T+1))LK_4\left(1 + \sup_{k\leq n(m)+k-1}\|x_k\|\right)\left(\sum_{i=0}^{k-1} a(n(m)+i)\left(\sum_{k=n(m)+i-\tau(n(m)+i)}^{n(m)+i-1} a(k)\right)\right)\end{aligned} \quad (2.82)$$

The stability of $\hat{x}(t)$ now follows analogously to the conclusion in Lemma 2.5. □

Proof of Corollary 2.9. Combine (2.76) and (2.78), then apply Assumption 2.1.5 and Lemma 2.8. □

Proof of Corollary 2.2. Under Assumption 2.1.1 - 2.1.5 it follows from Theorem 2.1 that x_n is almost surely stable. Using Lemma 2.10 it then follows that $x_{n+1} = x_n + a(n)[h(x_n) + e_n + M_{n+1}]$ with $e_n \in o(1)$. The convergence now follows from the SA literature; e.g., (V. Borkar 2022, Sec. 2.2). □

Proof of Theorem 2.12. Lemma 2.11 also holds for the sequence c_n with the limit h_∞, i.e., there exist $c_0 > 0$ and $T > 0$ such that for all initial conditions x on the closed unit ball, $\|\phi_{c_n}(x,t)\| < \frac{1}{2}$ for $t \in [T, T+1]$ and $c_n > c_0$. Now fix this $T > 0$ and follow the construction of the approximate T-length intervals as described at the beginning of Section 2.2. We will now define a new scaling sequence $s(m)$ to replace the scaling sequence in (2.12). Specifically,

$$s(m) := \max\{\inf_{n\geq 0}\{c_n \mid \sup_{l\leq m}\|\overline{x}(T_l)\| \leq c_n\}, 1\}. \quad (2.83)$$

Observe that $s(m)$ is again by construction monotonically increasing with the property that $s(m) \geq \|\overline{x}(T_m)\|$. It follows that the analysis presented in Section 2.2 holds for this new scaling sequence. It follows that the line of argument in Theorem 2.1 holds for the new scaling sequence with minor modifications, which we present for completeness.

Fix a sample point where the analogs of Lemma 2.6 and Lemma 2.10 hold for the new $T > 0$ and the new scaling sequence. Now suppose that $\sup_{m\geq 0}\|\overline{x}(T_m)\| < \infty$ does not hold, then $\sup_{l\leq m}\|\overline{x}(T_l)\| \nearrow \infty$. Observe that $s(m)$ is, in essence, a subsequence of c_n, where some elements get repeated. By construction, $s(m)$ thus inherits the convergence properties from $\{c_n\}$ and $h_{s(m)}(x) \to h_\infty(x)$ uniformly on compact sets. As before, let $x^m(t)$, $t \in [T_m, T_{m+1}]$, be the unique solution to the ODE $\dot{x}(t) = h_{s(m)}(x(t))$ with initial condition $x^m(T_m) = \hat{x}(T_m)$. The new

Lemma 2.11 thus shows there exists some $M > 0$ such that for all initial conditions on the closed unit ball, $\|x^m(t)\| < \frac{1}{2}$ for $t \in [T, T+1]$ and $m > M$.

Next, we again consider those timesteps where $s(m)$ increases, i.e., let $\{\tilde{m}(k)\}_{k\geq 1} \subset \{m\}_{m\geq 1}$, be those timesteps where $s(\tilde{m}(k)) > s(\tilde{m}(k)-1)$. In other words,

$$s(l) = s(\tilde{m}(k)) \tag{2.84}$$

for all $l \in \{\tilde{m}(k), \ldots, \tilde{m}(k+1)-1\}$. Consider now the first $\tilde{m}(k)$, such that $\tilde{m}(k) > M$ and $\sup_{t\in[T_m,T_{m+1}]}\|\hat{x}(t) - x^m(t)\| < \frac{1}{2}$ for $m > \tilde{m}(k)$ by the new Lemma 2.10. As before, the contradiction arises when we consider the last interval where $s(\tilde{m}(k))$ is used as a scaling factor. We have that

$$\frac{\overline{x}(T_{\tilde{m}(k+1)})}{s(\tilde{m}(k))} = \lim_{t\nearrow T_{\tilde{m}(k+1)}} \|\hat{x}(t)\| \leq \lim_{t\nearrow T_{\tilde{m}(k+1)}} \|x^{\tilde{m}(k+1)-1}(t)\| + \lim_{t\nearrow T_{\tilde{m}(k+1)}} \|\hat{x}(t) - x^{\tilde{m}(k+1)-1}(t)\| < 1 \tag{2.85}$$

This is a contradiction since $T_{\tilde{m}(k+1)}$ is by construction a timestep where $s(m)$ strictly increases, hence $\overline{x}(T_{m(k+1)}) > s(m(k))$. We conclude that $\sup_{m\geq 0} s(m) < \infty$ almost surely, and the theorem follows as before from Lemma 2.8. □

Proof of Corollary 2.13. We will verify *Assumption* 2.2.1. Pick any sequence $c_m \geq 1$ with $c_m \nearrow \infty$. The Lipschitz continuity of $h(x)$ yields that $\{h_{c_m}(x) : m \geq 0\}$ is an equicontinuous, pointwise bounded family of continuous functions. The Arzelà-Ascoli theorem (see A.1) thus shows that the family is relatively compact in the subspace consisting of continuous functions, equipped with the topology of compact convergence. In other words, there is a subsequence $m(n)$, such that $h_{c_{m(n)}}(x)$ converges to some limit $h_\infty(x)$ uniformly on compact sets. Define $c_n := c_{m(n)}$. It is left to show that $\dot{x}(t) = h_\infty(x(t))$ has the origin as its unique globally asymptotically stable equilibrium.

Condition (2.49) implies that there is some radius $r > 0$ and some $\varepsilon > 0$, such that $\lambda_{\max}(Dh(x)) < \varepsilon$ whenever $\|x\| > r$. Fix $x \in \mathbb{R}^d$ and some arbitrary $y \in \mathbb{R}^d$ with $\|y\| > r$. Now apply the gradient theorem for Lipschitz continuous functions (see A.1) to every coordinate of h, then

$$\frac{h(c_n x)}{c_n} - \frac{h(y)}{c_n} = \left(\int_0^1 Dh(y + t(c_n x - y))dt\right)\frac{(c_n x - y)}{c_n}. \tag{2.86}$$

Thus, $h_\infty(x) = \lim_{n\to\infty}\left(\int_0^1 Dh(y + t(c_n x - y))dt\right)x$ with $\lambda_{\max}(Dh(y + t(c_n x - y))) < \varepsilon$ for all t, n. Using basic calculus and linear algebra, we conclude that for every $x \in \mathbb{R}^d$, $h_\infty(x) = \mathbf{A}_\infty(x)x$ for some matrix $\mathbf{A}_\infty(x) \in \mathbb{R}^{d\times d}$ with $\lambda_{\max}(\mathbf{A}_\infty(x)) < \varepsilon$. Thus $\dot{x}(t) = h_\infty(x(t))$ is globally asymptotically stable to the origin by LaSalle's invariance principle; see, e.g., (V. Borkar 2022, Appendix B). □

Proof of Lemma 2.16. We shall only discuss a proof for $q = 1$ to simplify the presentation. Recall that the heavy ball iteration is

$$x_{n+1} = x_n + a(n)(1-\beta)\left[\sum_{i=0}^{n} \beta^{n-i} g(x_i)\right], \tag{2.87}$$

with $g(x_n) \coloneqq h(x_n) + M_{n+1}$. Then using Gronwalls inequality (see A.1), the linear growth of h, $\mathbb{E}\left[\|M_{n+1}\|^2\right] \leq K^2(1 + \mathbb{E}\left[\|x_n\|^2\right])$, and Assumption 2.1.4 it follows that

$$\mathbb{E}\left[\|x_n\|^2\right] \leq Cn^2 \tag{2.88}$$

for some constant $C > 0$. With this, it follows from the martingale convergence theorem (see A.2) that $\sum_{n\geq 0} a(n)\frac{M_{n+1}}{n}$ converges almost surely. A second use of Gronwalls inequality then shows that $\|x_n\| \in \mathcal{O}(n)$. Finally, we conclude that

$$\|\delta_n\| \in \mathcal{O}\left(\sum_{i=0}^{n-\tau(n)-1} \beta^{n-i}(1+i) + \|\sum_{i=0}^{n-\tau(n)-1} \beta^{n-i} M_{i+1}\|\right)$$

With $\tau(n) \nearrow \infty$, the lemma follows after a small calculation. □

Proof of Theorem 2.14. First, observe that $\tilde{M}_{n+1}$ is also a zero-mean martingale difference sequence with respect to $\mathcal{F}_n \coloneqq \sigma(x_0, M_1, \ldots, M_n), n \geq 0$. Using the Lipschitz continuity of h, it follows that

$$\|e_n\| \leq L \sum_{i=n-\tau(n)}^{n-1} \beta^{n-i}\|x_n - x_i\|. \tag{2.89}$$

Analogously to Section 2.2, we can now derive recursive inequalities in L_2 and in norm. We will illustrate this for the L_2 case. Define $\tilde{e}_n \coloneqq \sum_{i=n-\tau(n)}^{n-1} \beta^{n-i}\|x_n - x_i\|$.

Using a telescoping sum, (2.57) and $\|h(x)\| \leq K(1 + \|x\|$ we have for $n \geq i$ that

$$\|x_n - x_i\| \leq \sum_{j=i}^{n-1} \tilde{a}(j)K(1 + \|x_j\|) + \|\sum_{j=i}^{n-1} \tilde{a}(j)\tilde{M}_{j+1}\| + \sum_{j=i}^{n-1} \tilde{a}(j)L\tilde{e}_j + \sum_{j=i}^{n-1} \tilde{a}(j)\|\delta_j\|. \tag{2.90}$$

Since $\mathbb{E}\left[\|M_{n+1}\|^2\right]^{\frac{1}{2}} \leq K(1 + \mathbb{E}\left[\|x_n\|^2\right]^{\frac{1}{2}})$, the new martingale noise sequence satisfies

$$\mathbb{E}\left[\|\tilde{M}_{n+1}\|^2\right]^{\frac{1}{2}} \leq \frac{1}{c(n)} \sum_{i=n-\tau(n)}^{n} \beta^{n-i}\mathbb{E}\left[\|\tilde{M}_{i+1}\|^2\right]^{\frac{1}{2}} \tag{2.91}$$

$$\leq \frac{K}{1-\beta}\left(1 + \mathbb{E}\left[\|x_n\|^2\right]^{\frac{1}{2}} + \sum_{i=j-\tau(j)}^{j-1} \beta^{j-i}\mathbb{E}\left[\|x_j - x_i\|^2\right]^{\frac{1}{2}}\right), \tag{2.92}$$

where we used (2.54). Now define the L_2 error sequence $\tilde{e}_n^{L_2} \coloneqq \sum_{i=n-\tau(n)}^{n-1} \beta^{n-i}\mathbb{E}\left[\|x_n - x_i\|^2\right]^{\frac{1}{2}}$, where by (2.92) each term satisfies

$$\mathbb{E}\left[\|x_n - x_i\|^2\right]^{\frac{1}{2}} \leq \sum_{j=i}^{n-1} \tilde{a}(j)K_1(1 + \mathbb{E}\left[\|x_j\|^2\right]^{\frac{1}{2}}) + \sum_{j=i}^{n-1} \tilde{a}(j)K_2\tilde{e}_j^{L_2} + \sum_{j=i}^{n-1} \tilde{a}(j)\mathbb{E}\left[\|\delta_j\|^2\right]^{\frac{1}{2}}. \tag{2.93}$$

for some constants $K_1, K_2 > 0$. Thus, using (2.54), we have that

$$\tilde{e}_n^{L_2} \leq \frac{K_1}{1-\beta}(1 + \sup_{j \leq n-1} \mathbb{E}\left[\|x_j\|^2\right]^{\frac{1}{2}}) \left(\sum_{j=n-\tau(n)}^{n-1} \tilde{a}(j) \right) + \frac{K_2}{1-\beta} \left(\sum_{j=n-\tau(n)}^{n-1} \tilde{a}(j) \tilde{e}_j^{L_2} \right) + \frac{1}{1-\beta} \left(\sum_{j=n-\tau(n)}^{n-1} \tilde{a}(j) \mathbb{E}\left[\|\delta_j\|^2\right]^{\frac{1}{2}} \right). \tag{2.94}$$

The similarity to the recursive inequality (2.35) should be clear. The same inequality will hold when the sequences are scaled by the monotone scaling sequence used in Section 2.2. Notably, we do not need to use Egorov's Theorem here since the AoI sequence is deterministic. We can thus apply the new Gronwall-type inequality Lemma 2.3. As $\mathbb{E}\left[\|\tilde{e}_n\|^2\right]^{\frac{1}{2}} \leq \tilde{e}_n^{L_2}$, evaluating the recursion for $\tilde{e}_n^{L_2}$ leads to the required L_2 bound for $\tilde{e}_n$. Furthermore, from (2.90), we also arrive at a recursive inequality for $\tilde{e}_n$ itself. The stability analysis in Section 2.2 then goes through using the L_2 bound and the norm bound for $\tilde{e}_n$ along exactly the same line of argument. The tail error δ_n only adds neglectable terms in the analysis. We omit the details to avoid redundancies. □

Proof of Corollary 2.15. Theorem 2.14 shows that there is sample path dependent radius $R > 0$, such that $x_n \in \mathcal{B}_R(0)$ for all $n \geq 0$. It thus follows that $\tilde{e}_n \leq \frac{2B}{1-\beta}$. Further, using the martingale convergence theorem yields that $\sum_{j=1}^{n} \tilde{a}(j)\tilde{M}_{j+1}$ converges almost surely. Assumption 2.1.5 then implies that $\sum_{j=i}^{n-1} \tilde{a}(j) \to 0$ for all $n - \tau(n) \leq i \leq n-1$. It then follows that the right-hand side of (2.90) converges to zero almost surely and thus (2.89) shows that $\|e_n\| \to 0$ almost surely. Equation (2.57) is, therefore, again a standard SA iteration with vanishing additive error, and the corollary follows. □

Chapter 3

Distributed Asynchronous Stochastic Gradient Descent Methods

As a first application of the theory developed in Chapter 2, we consider DASGD applied to a decentralized optimization problem as sketched in Example 1.0.1. As mentioned in the introduction, the most common examples of this class of problems are ERM problems over deep neural networks.

Consider the unconstrained stochastic optimization problem $\min_{x\in\mathbb{R}^d} F(x)$ with objective

$$F(x) \coloneqq \int_{\Xi} f(x;\xi) d\mathbb{P}(\xi) \tag{3.1}$$

for some random function $f : \mathbb{R}^d \times \Xi \to \mathbb{R}$.

Suppose the global optimization variable $x = (x_1, \ldots, x_D) \in \mathbb{R}^d$, with $d \coloneqq \sum_{i=1}^D d_i$, is the concatenation of local variables x_i, where a set of workers controls each local variable. To solve the problem, systems calculate sample partial derivatives $\nabla_{x_i} f(\cdot, \xi)$ to adapt their associated local variable. We assume that the systems do not know the distribution of ξ, but for the computation of every sample partial derivative, use a local i.i.d. realization ξ_n^i of ξ. Further, suppose the systems work asynchronously and are potentially physically distributed. Together, asynchronous computing and communication will then induce AoI sequences $\tau_{ij}(n)$ for the component iterations of the updated global variable. The complete algorithm is presented in Algorithm 1. Each local variable is then effectively updated by the following asynchronous distributed SGD iteration starting from some initial value $x_0^i \in \mathbb{R}^{d_i}$:

$$x_{n+1}^i = x_n^i - a(n)\nabla_{x_i} f(x^1_{n-\tau_{1i}(n)}, \ldots, x^i_{n-\tau_{ii}(n)}, \ldots, x^D_{n-\tau_{Di}(n)}; \xi_n^i). \tag{3.2}$$

We will now formulate conditions to apply the theory developed in Chapter 2.

Algorithm 1 Asychronous Distributed SGD Algorithm with D components

Initialize the master iteration with $x_0 \in \mathbb{R}^d$ and a stepsize $a(n) > 0$ for all $n \geq 0$.
for the entire duration **do**
  for each worker associated with component i **do**
    Receive an SGD job (master iterate and a data sample).
    Wait for the completion of other jobs in the worker queue.
    Compute stochastic gradient.
    Send computed gradient to the master of component i.
  for the master of component i **do**
    Receive stochastic gradient $\nabla_{x^i} f(x', \xi')$ from some worker.
    Update iterate: $x^i \leftarrow x^i - a(n^i)\nabla_{x^i} f(x', \xi')$.
    Request current iterate from other masters.
    Send the worker a job (sample iterate x' and a new data sample ξ').
    $n^i \leftarrow n^i + 1$.

3.1 Assumptions and main statements

Denote the Hessian matrix of f with respect to x by $\mathbf{H}_x f(x;\xi)$. Further, $\|\mathbf{H}_x f(x;\xi)\|$ denotes the induced matrix norm on $\mathbb{R}^d$ and $\lambda_{\min}(\cdot)$ denotes the smallest eigenvalue of some matrix.

Assumption 3.1.1.

1) $f(x;\xi)$ *is twice differentiable in* x *for* $\mathbb{P}$*-almost all* ξ*,*
2) $\sup_x \|\mathbb{E}\left[\mathbf{H}_x f(x;\xi)\right]\| \coloneqq L < \infty$*,*
3) $\mathbb{E}\left[\|\nabla_x f(x;\xi)\|^2\right] \leq K(1 + \|x\|^2)$ *for all* $x \in \mathbb{R}^d$ *for some* $K > 0$*.*
4) $\liminf_{\|x\|\to\infty} \lambda_{\min}\left(\mathbb{E}\left[\mathbf{H}_x f(x;\xi)\right]\right) > 0$*,*

Theorem 3.1. *Under Assumption 2.1.4, 2.1.5 and 3.1.1 the DSGD iterations (3.2) converge almost surely to a limit point* x_∞*, such that* $\nabla_x F(x_\infty) = 0$*, i.e. a stationary point of (3.1).*

For Assumption 3.1.1, we have that each of the following two conditions in addition to Assumption 3.1.1.1 and 3.1.1.2 imply Assumption 3.1.1.3:

1) ξ has finite support, i.e., $|\Xi| < \infty$, which is typically for machine learning applications.
2) $\|\nabla_x f(x;\xi)\| \leq |g(\xi)|(1 + \|x\|)$ for some measurable function $g : \Xi \to \mathbb{R}$, such that $\mathbb{E}\left[g(\xi)^2\right] < \infty$.

Observe that Assumption 3.1.1.3 allows that the gradient $\nabla_x F(x)$ is unbounded. Such objectives are not covered by the unconstrained optimization setting considered in (Zhou et al. 2022). Assumption 3.1.1.4 is the stability condition to apply the theory developed in Chapter 2. Useful conditions that imply Assumption 3.1.1.4 are:

1) $F(x) = G(x) + \kappa\|x\|^2$ for any L_2 regularization coefficient $\kappa > 0$, where $G(x)$ satisfies Assumption 3.1.1 and that $G(x) \in o(\|x\|^2)$.
2) $F(x) = G(x) + L\|x\|^2$, where $G(x)$ satisfies Assumption 3.1.1 with $L > 0$ and $G(x)$ is coercive, i.e., $G(x) \to \infty$ as $\|x\| \to \infty$

Remark 3.1.1. *With the insights from Remark 2.2.3, we can further weaken Assumption 3.1.1.4 and merely require that* $\liminf_{\|x\|\to\infty} \lambda_{\min}\left(\mathbb{E}\left[\mathbf{H}_x f(x;\xi)\right]\right) \geq 0$. *This asymptotic positive-semi definiteness condition of the expected Hessian holds whenever $F(x)$ is coercive, which can be shown using the mean value theorem for vector-valued functions (McLeod 1965).*

An important class of problems that becomes available for distributed SGD under large delays are quadratic objectives:

Example 3.1.1. *Consider*

$$f(x;\xi) = x^\top A(\xi)x + b(\xi)^\top x, \tag{3.3}$$

where $\mathbb{E}\left[A(\xi)\right] \in \mathbb{R}^{d\times d}$ *is a positive definite matrix with* $\mathbb{E}\left[\|A(\xi)\|^2\right] < \infty$ *and* $\mathbb{E}\left[\|b(\xi)\|^2\right] < \infty$. *Assumption 3.1.1 holds for (3.3) and Theorem 3.1 thus shows (3.2) converges to the global minimum of* $F(x) = x^\top \mathbb{E}\left[A(\xi)\right] x + \mathbb{E}\left[b(\xi)\right]^\top x$.

Remark 3.1.2. *Assumption 3.1.1.3, implies that $\nabla_x f(x;\xi)$ is uniformly integrable for every open neighborhood of some $x \in \mathbb{R}^d$. It thus follows from Lebesgue's dominated convergence theorem that the sample gradients are unbiased (Barakat and P. Bianchi 2021), i.e.,* $\mathbb{E}\left[\nabla_x f(x;\xi)\right] = \nabla_x \mathbb{E}\left[f(x;\xi)\right] = \nabla_x F(x)$ *with objective F from (3.1).*

3.2 Analysis

We apply the theory developed in Chapter 2. Define $h(x) := -\nabla_x F(x)$ and $h_c(x) := -\frac{\nabla_x F(cx)}{c}$. Further, by Assumption 3.1.1.1 and local approximation of the Hessian using gradients, Assumption 3.1.1.3 also implies that $\mathbf{H}_x f(x;\xi)$ is uniformly integrable for open neighborhoods, such that $\mathbb{E}\left[\mathbf{H}_x f(x;\xi)\right] = \mathbf{H}_x F(x)$. Assumption 3.1.1 now implies three simple auxiliary Lemmas.

Lemma 3.2. *$F(x)$ is coercive.*

Proof. The Lemma follows from Assumption 3.1.1.4 by applying the mean value theorem for vector-valued functions, (McLeod 1965), two times for $F(x)$ and $\nabla_x F(x)$ to obtain a quadratic form for $h(x)$. □

Lemma 3.3. *$h(x)$ is Lipschitz-continuous, with Lipschitz constant $L > 0$ from Assumption 3.1.1.2.*

Proof. The lemma follows directly from Assumption 3.1.1.2 and the mean value theorem. □

Lemma 3.4. $\limsup_{\|x\|\to\infty} \lambda_{\max}(Dh(x)) < 0$

Proof. The lemma follows directly from Assumption 3.1.1.4. □

Proof of Theorem 3.1. Rewrite the SGD iterations as

$$x^i_{n+1} = x^i_n + a(n)[h(x^1_{n-\tau_{i1}(n)}, \dots, x^D_{n-\tau_{iD}(n)}) + M^i_{n+1}],$$

with $M^i_{n+1} \coloneqq \nabla_x F(x^1_{n-\tau_{i1}(n)}, \dots, x^D_{n-\tau_{i1}(n)}) - \nabla_{x_i} f(x^1_{n-\tau_{i1}(n)}, \dots, x^D_{n-\tau_{i1}(n)}, \xi^i_n)$. Lemma 3.3 shows that $h(x)$ Lipschitz continuous and thus satisfies Assumption 2.1.1. Next, since ξ^i_n are i.i.d., Assumption 3.1.1(c) yields that Assumption 2.1.3 holds for M_{n+1}. Finally, Lemma 3.4 shows that the condition in Corollary 2.13 holds. Hence, Corollary 2.13 shows that x_n is stable almost surely and converges almost surely to a potential sample path-dependent compact connected internally chain transitive invariant set of the ODE $\dot{x}(t) = h(x(t))$. By Lemma 3.2, F is coercive and thus a Lyapunov function for $\dot{x}(t) = h(x(t))$, hence the only possible invariant set is the set of critical points of F, which is non-empty as F is coercive. □

3.2.1 Rate of convergence

Next, we are interested in the almost sure rate of convergence of DASGD as a function of AoI moment bounds. To analyze the rate of convergence of DASGD, we use a slightly stronger version of Assumption 3.1.1.2.

Assumption 3.2.1.

1) *$f(x;\xi)$ is twice continuously differentiable in x for $\mathbb{P}$-almost all ξ, further $\nabla_x f(x;\xi)$ and $\mathbf{H}_x f(x;\xi)$ are continuous in ξ.*
2) $\sup_x \|\mathbf{H}_x f(x;\xi)\| \coloneqq L < \infty$.
3) *The sample space Ξ is compact.*
4) $\liminf_{\|x\|\to\infty} \lambda_{\min}(\mathbb{E}[\mathbf{H}_x f(x;\xi)]) > 0$.

Objective functions parameterized by neural networks with Gaussian error linear units (GELUs) are the prime examples that satisfy Assumption 3.2.1.1 (Hendrycks and Gimpel 2016). Thus, Assumption 3.2.1.1 is no restriction. For more discussion on GELUs, we refer to Chapter 4. Further, for most applications, datasets are given and preprocessed, guaranteeing that the sample space Ξ is compact or even finite.

From Assumption 3.2.1 it follows that Assumption 3.1.1 holds and moreover that

$$\sup_{x,\xi} \|\mathbf{H}_x f(x;\xi)\| < \infty, \tag{3.4}$$

hence

$$\|\nabla_x f(x;\xi) - \nabla_x f(y;\xi)\| \leq L'\|x-y\| \text{ for all } x,y \in \mathbb{R}^d \text{ and } \xi \in \Xi \tag{3.5}$$

for some $L' > 0$. This point-wise Lipschitz inequality enables a straightforward rate of convergence analyses of (3.2) using the obtained stability from (2) and telescoping, which is a classical tool in gradient and stochastic gradient descent analysis (Goodfellow, Bengio, and Courville 2016).

Theorem 3.5. *Under Assumption 2.1.4, 2.1.5, 3.2.1 the DASGD iterations (3.2) converge almost surely to the set of stationary points with almost sure rate:*

$$\min_{k=0,\dots,n} \|\nabla_x F(x_k)\|_2^2 \in \mathcal{O}\left(\frac{1 + \sum_{i,j}\sum_{k=0}^{n} a(k) \sum_{m=k-\tau_{ij}(k)}^{k-1} a(m)}{\sum_{k=0}^{n} a(k)}\right). \tag{3.6}$$

Theorem 3.5 shows that the almost sure rate is determined by the convergence rate of $\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)$, which was the key sufficient condition for the distributed BMT developed in Chapter 2. In other words, its convergence determines both stability and the convergence rate. We are now interested in optimizing the convergence rate by suitable AoI-dependent stepsize choice.

3.2.2 A rate optimal stepsize rule from AoI moment bounds

Based on (3.6), we now present a rate optimal stepsize rule. The theorem is stated at this point of the thesis due to its close connection to Theorem 3.5. However, the proof will be discussed in Chapter 5, after we establish a better understanding $\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k)$. We state the theorem by using stochastic dominance as defined in Definition 0.0.6.

Theorem 3.6. *Consider the setting of Theorem 3.5, such that all $\tau_{ij}(n)$ are stochastically dominated by random variable τ with $\mathbb{E}[\tau^p] < \infty$, $p > 0$. Further, apply the stepsize rule:*

$$a(n) = \frac{c}{(n+1)^q} \text{ with } q := \min\left\{\frac{1}{2}\left(1+\frac{1}{p}\right), 1\right\} \text{ and } c > 0 \text{ for } p > 1. \tag{3.7}$$

If $p > 1$, then

$$\min_{k=0,\dots,n} \|\nabla_x F(x_k)\|_2^2 \in \mathcal{O}(n^{-\frac{1}{2}(\frac{p-1}{p})}), \tag{3.8}$$

else:

$$\min_{k=0,\dots,n} \|\nabla_x F(x_k)\|_2^2 \in o(1). \tag{3.9}$$

Remark 3.2.1. *For $p \in (0,1]$, we can pick $a(n) = \frac{c}{(n+1)\log(n+2)}$ to obtain at least a slow convergence rate in $\mathcal{O}\left(\frac{1}{\log(\log(n))}\right)$. Practically, we observed that it is better to use $a(n) = \frac{c}{(n+1)}$*

for $p \in (0,1]$, though it theoretically only guarantees $\min_{k=0,\dots,n} \|\nabla_x F(x_k)\|_2^2 \in o(1)$. We expect that it is possible to select a better stepsize for $p \in (0,1]$ to obtain a rate better than $\mathcal{O}\left(\frac{1}{\log(\log(n))}\right)$, but worse than $\mathcal{O}\left(\frac{1}{\log(n)}\right)$, since we at least have to choose $a(n) \in \mathcal{O}\left(\frac{1}{n}\right)$ to ensure stability.

Theorem 3.6 provides a strong characterization that with probability one, all trajectories of DASGD (3.2) converge to a stationary point. Further, the stepsize is chosen for $p > 1$ to optimize the convergence rate. In an asynchronous computing setting, the "slowest" system will then give rise to the convergence rate as it leads to the smallest p to obtain a moment bound. In general, more heavy-tailed traffic slows down the convergence rate, and for less heavy-tailed traffic, we have that as $p \to \infty$ the rate approaches the almost sure rate of SGD (Jun Liu and Yuan 2022). Next, numerical experiments are present that verify the stepsize recipe from Theorem 3.6.

3.3 Numerical verification

We simulate a distributed computing scenario to verify the stepsize rule proposed in Theorem 3.6. Specifically, we consider $K = 100$ systems with heterogenous processing times that follow a Pareto distribution with exponents in $p \in [2,6]$, implying that the p-th moment of the processing times is bounded (Cinlar and ðCınlar 2011). One system is chosen with $p = 2$, while all other systems get a Pareto exponent chosen uniformly at random from $[2,6]$.

For the optimization problem, we consider an approximation problem where a quadratic function $g : \mathbb{R}^2 \to \mathbb{R}$ should be approximated on the subset $[0,1]^2 \subset \mathbb{R}^2$ by a linear map

$$L_x : \mathbb{R}^2 \to \mathbb{R}, \quad z \mapsto \begin{bmatrix} x^1, x^2 \end{bmatrix} \begin{bmatrix} z^1 \\ z^2 \end{bmatrix} + x^3, \tag{3.10}$$

with parameter vector $x = (x^1, x^2, x^3) \in \mathbb{R}^3$. The function g is given as a black box, and only input-output tuples $(z, g(z))$ can be sampled. Using Algorithm 1, we are looking for the linear map that minimizes

$$F(x) \coloneqq \mathbb{E}_z (L_x(z) - g(z))^2, \tag{3.11}$$

where z are sampled uniformly from $[0,1]^2$. For the computing systems, we let three-fourths of the systems update (x^1, x^2) and the other one-fourth update x^3. This adds additional inhomogeneity and emulates a non-uniform assignment of workers to subspaces of an optimization problem.

For the simulation, we focus on the polynomial the stepsizes $a(n) = \frac{1}{(n+1)^q}$ and compare:

1) The largest possible (slowest decaying) stepsize to guarantee stability and convergence, i.e., $q_{max} = \frac{1}{2} + \delta$ according to Theorem 3.1, for some arbitrary small $\delta > 0$.

2) The smallest possible (fastest decaying) stepsize, i.e., $q_{\min} = 1$, such that Assumption 2.1.4 still holds.
3) The proposed optimal stepsize rule from Theorem 3.6, i.e. $q_{\text{opt}} = \frac{1}{2}\left(1 + \frac{1}{p}\right) = \frac{3}{4}$.

The simulation then runs Algorithm 1 for all three cases. It is easy to see that the assumptions of Theorem 3.6 are satisfied, which therefore predicts that q_{opt} should guarantee the fastest convergence of $\|\nabla_x F(x_n)\|$ to zero. As $F(x)$ is convex, the limit is the global minimum and, thus, the optimal linear approximation.

Figure 3.1 and Figure 3.2 show the resulting comparison of the three runs. During simulation, $\|\nabla_x F(x_n)\|$ is evaluated by numerical integration. Figure 3.1 plots the rolling average with window length 1000 to better capture the convergence rate. Figure 3.2 plots the raw data. We see that q_{opt} indeed provides the best asymptotic convergence rate. On the other hand, $q_{\max}$ can at least provide a good approximation in finite time, but the initial rapid progress with $q_{\max}$ stagnates after 10000 steps, where it has not yet decayed sufficiently to compensate for the effect of the AoI. This lack of compensation to counter AoI is especially visible in Figure 3.2, which shows the high variance of the optimization run with $q_{\max}$. Still, the effect of the AoI is also clearly visible for q_{opt}, for which we can observe better AoI compensation even in a close neighborhood to the optimal value. The aforementioned properties can be expected to be even more severe when Algorithm 1 is applied to large-scale non-convex machine learning problems run on parallel computing infrastructure. In such cases, choosing the right stepsize will be essential to accelerate the training. Theorem 3.6 thus provides important guidelines for stepsize selection. Combined with the results to be presented Chapter 7 on predicting AoI from processing time data, we have therefore established a fundamental framework to connect processing time data with convergence rate prediction - the core answer to the second part of (Q2).

3.4 Discussion and related work

Various distributed and asynchronous implementations of SGD have been proposed throughout the last two decades. The initial rise of deep learning came with GPUs, which made training modestly-sized neural networks practical (Raina, Madhavan, and Ng 2009). The GPU acts as thousands of smaller CPUs working in parallel on a subspace of the optimization problem. However, this approach is limited to training models that fit into GPU memory. The same idea has been used to divide the optimization space among multiple computing nodes, known as model parallelism (Krizhevsky, Hinton, et al. 2009). Conventionally, model parallel synchronous SGD training has been implemented as a synchronous algorithm, where all coordinate gradients are collected from all systems to apply a complete sample gradient. When waiting for other systems is omitted and processing does continue, we arrive at DASGD with a single system for each

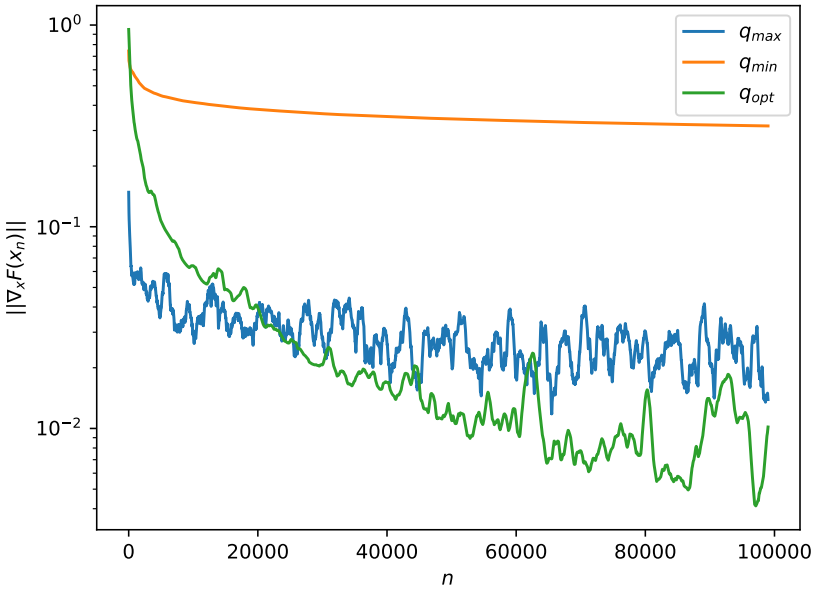

Figure 3.1: Comparison of three stepsize schedules for the proposed linear approximation problem. The graphs show the rolling average with a window of length 1000.

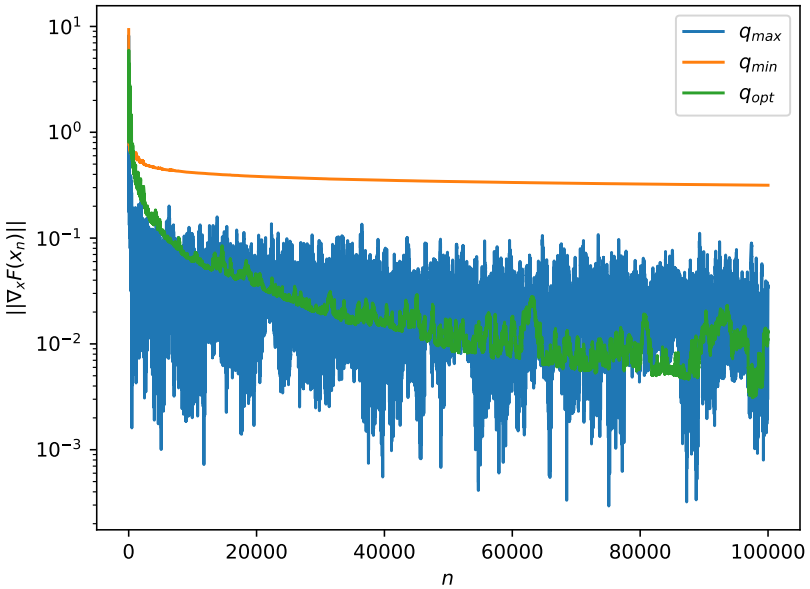

Figure 3.2: Comparison of three stepsize schedules for the proposed linear approximation problem. The graphs show the raw data of Figure 3.1.

subspace. This is also known as the inconsistent write/read model (Ji Liu and Wright 2015).

In (Dean et al. 2012), the authors proposed downpour SGD, which implements a version of the aforementioned synchronous SGD several times in parallel to update a single model asynchronously. This is known as data parallelism. In the same work, practically less-than-ideal speedups have been reported for constant stepsizes due to varying processing times across different systems, leading to many systems waiting for the slowest system in a synchronous SGD run to finish a given phase of computation (Dean et al. 2012). When the waiting in downpour SGD is omitted, we arrive at the full DASGD algorithm. It is shown in (Lian et al. 2015) that even when the AoI is bounded, ASGD with constant stepsize $a > 0$ can only be guaranteed to converge to a neighborhood of a stationary point with radius in $\mathcal{O}(a^2T^2)$, where $T > 0$ is the AoI bound. Here, the neighborhood is a function of the objectives' smoothness and a bound on the sampling variance. This information is not available without estimation during runtime for typical deep-learning problems. In other words, the stepsize can not be chosen to guarantee a small neighborhood. More importantly, the distance to a stationary point may become significant when the number of systems increases since T typically is in the order of the number of systems. We conclude that constant stepsizes are unsuitable for scaling DASGD to a large, potentially varying number of systems without overheads for synchronization.

In the decreasing stepsize regime, two recent developments for DASGD with large, unbounded AoI exist. Zhou et al. (2022) show that fast decaying stepsizes have to be chosen to counter large unbounded deterministic delays. The assumption of deterministic delays is crucial in this analysis, which only covers data parallelism. As we saw, we instead concluded the results from Theorem 2.1 on distributed stochastic approximation algorithms under stochastic AoI. The connection between processing time distributions and AoI will be further discussed once we establish the AoI in distributed computing modeled as parallel point processes in Chapter 7.

Distributed SGD with bounded information delays was first considered in (J. Tsitsiklis, D. Bertsekas, and Athans 1986). Here, it was sketched for the first time that delays may be allowed to grow sublinearly relative to a global clock when a sufficiently rapid decaying stepsize is chosen. Finite time error bounds for asynchronous SGD algorithms under convex stochastic objectives, constant stepsizes, and bounded delays were proposed in (Agarwal and Duchi 2011) and (Feyzmahdavian, Aytekin, and Johansson 2016). Finite time bounds for the mean square variation of the mean gradient of SGD under a time-varying stepsize were proposed in (Lian et al. 2015) for general non-convex objectives and bounded delays. Almost sure convergence of SGD to stationary points under merely locally Lipschitz continuous gradients with noise-dependent Lipschitz constants was proven in (Ramaswamy, Redder, and Daniel E Quevedo 2021a). However, stability and the delay conditions proposed in (V. Borkar 2022, Chapter 6) were assumed. Various finite-time, mean-square-error bounds for ASGD algorithms have been proposed throughout

the last decades for bounded delays Agarwal and Duchi 2011; Feyzmahdavian, Aytekin, and Johansson 2016; Lian et al. 2015. We refer to Mishchenko et al. 2022; Anastasiia Koloskova, Stich, and Jaggi 2022 for the most advanced mean-square-error bounds for delay-adaptive ASGD versions that assume AoI in the order of the number of workers. Instead, this chapter focuses on almost sure asymptotic convergence rate estimates for the complete DASGD version.

Regarding delays, the closest to the present chapter is (Zhou et al. 2022). The delays considered therein are potentially large and unbounded but are assumed to be deterministic. We claim that it is more representative to work with stochastic delays, which complicates the analysis significantly, as we saw in Chapter 2. The algorithm considered in (Zhou et al. 2022) is SGD, where multiple nodes compute updates for a single global variable. The authors focus on two scenarios: general non-convex objectives and variational coherent objectives. The authors use projected asynchronous SGD in the second scenario to ensure stability and a potential compact convex constraint. Then, an elegant analysis based on energy functions is used to ensure global almost sure convergence. The second scenario is not the scope of this work, and we instead compare it to the first scenario, which considers asynchronous SGD without projections for an unconstrained non-convex optimization problem with objective $F(x) \coloneqq \mathbb{E}_\xi[f(x;\xi)]$. The paper shows that $\lim_{n\to\infty} \mathbb{E}\left[\|\nabla_x F(x_n)\|^2\right] = 0$, i.e., the gradient converges in mean square, where here the expectation is with respect to x_n. We provide conditions showing $\|\nabla_x F(x_n)\| \to 0$ a.s. at an AoI-dependent almost rate. This provides a strong characterization for every individual trajectory of the algorithm and shows that practically every instantiation of the algorithm converges to a critical point. Furthermore, the presented analysis holds under weaker assumptions. Both analyses require that $\mathbb{E}_\xi[\nabla_x f(x;\xi)]$ is Lipschitz continuous. The analysis in (Zhou et al. 2022) requires that $\sup_{x\in\mathbb{R}^d} \mathbb{E}_\xi[\|\nabla_x f(x;\xi)\|^2] < \infty$. We only require that $\mathbb{E}_\xi[\|\nabla_x f(x;\xi)\|^2] \leq K(1 + \|x\|^2)$ for all $x \in \mathbb{R}^d$ for some $K > 0$ and we thus even allow the objective gradient $\nabla_x F(x)$ to be unbounded. In addition, using the insights from remark 2.2.3, we merely require that $F(x)$ is coercive to apply the established distributed BMT.

In summary, the analysis covers for the first time the stability of DASGD for a wide class of objectives in the presence of large unbounded stochastic delays without bounded first moment. Further, with Theorem 3.6, we provide an AoI-dependent convergence rate that was verified by numerical examples in the previous section. Hence, this provides the core answer on the convergence rate raised in (Q2). In Table 3.1 below, we summarize the progress in the literature on the asymptotic convergence analysis of DASGD. We concentrate on the most relevant previous works with unbounded delays. As mentioned before, there is a wide literature on delay-adaptive ASGD methods on a single variable iteration with bounded AoI, which we do not consider here.

	Stability	a.s.	Unb. AoI	AoI $p \in [0,1]$	Unb. drift	Distributed	Rate
[1]	✗	✓	✓	✗ ($p > 1$)	✓	✓	✗
[2]	✓	✗	✓	✗ (determ.)	✗	✗	✗
[3]	✓	✗	✓	✗ ($p > 2$)	✓	✗	✓
This	✓	✓	✓	✓	✓	✓	✓

Table 3.1: Asymptotic convergence results for ASGD and DASGD. [1] – (Ramaswamy, Redder, and Daniel E Quevedo 2021a), [2] – Zhou et al. 2022, [3] – X. Zhang, Jia Liu, and Zhu 2020.

3.5 Proofs of Chapter 3

Proof of Theorem 3.5. The proof is instructive since we already know that x_n is stable and convergent almost surely from Theorem 3.1. The proof, in essence, uses a standard telescoping sum approach. To streamline the presentation, we concentrate on the single coordinate iteration:

$$x_{n+1} = x_n - a(n)\nabla_x f(x_{n-\tau(n)}; \xi_n). \tag{3.12}$$

First, recall the effect of the AoI process. The errors due AoI that arise at each step in (3.2) are:

$$\|\nabla_x f(x_{n-\tau(n)}; \xi_n) - \nabla_x f(x_n; \xi_n)\| \leq L' \|x_{n-\tau(n)} - x_n\| \leq L' \sum_{k=n-\tau(n)}^{n-1} \|x_{k+1} - x_k\| \tag{3.13}$$

$$\leq L' \sum_{k=n-\tau(n)}^{n-1} a(k) \|\nabla_x f(x_{k-\tau(k)}; \xi_k)\|, \tag{3.14}$$

with L' from (3.5). Using Assumption 3.2.1 and the stability of x_n from Theorem 3.1 it follows that

$$\|\nabla_x f(x_{n-\tau(n)}; \xi_n) - \nabla_x f(x_n; \xi_n)\| \leq \mathcal{O}\left(\sum_{k=n-\tau(n)}^{n-1} a(k) \right). \tag{3.15}$$

Next, with the Lipschitz continuity of $\nabla_x F(x)$ it follows that

$$F(x_{n+1}) \leq F(x_n) + \langle \nabla_x F(x_n), x_{n+1} - x_n \rangle + \frac{L}{2} \|x_{n+1} - x_n\|^2, \tag{3.16}$$

where $\langle \cdot, \cdot \rangle$ is the dot product and for this proof $\|\cdot\|$ is the Euclidean norm. See (Y. Nesterov 2003, Lemma 1.2.3) for a proof of (3.16).

Next, add to zeros and write (3.12) as

$$x_{n+1} - x_n = a(n)\nabla_x f(x_{n-\tau(n)}; \xi_n) = a(n) \left(\nabla_x F(x_n) + e_n + M_{n+1} \right), \tag{3.17}$$

with $e_n \coloneqq \nabla_x f(x_{n-\tau(n)}; \xi_n) - \nabla_x f(x_n; \xi_n)$ and $M_{n+1} \coloneqq \nabla_x f(x_n; \xi_n) - \nabla_x F(x_n)$. As before, M_{n+1} is the zero mean Martingale difference sequence as ξ_n are i.i.d., and e_n is the error due to

AoI at the n-th master iteration step. Then, using $\|a+b+c\|^2 \leq 4(\|a\|_2^2 + \|b\|^2 + \|c\|^2)$, (3.16) yields that

$$\begin{aligned} F(x_{n+1}) \leq F(x_n) - a(n)\|\nabla_x F(x_n)\|^2 - a(n)\langle \nabla_x F(x_n), e_n\rangle - a(n)\langle \nabla_x F(x_n), M_{n+1}\rangle \\ + a(n)^2 2L\left(\|\nabla_x F(x_n)\|^2 + \|e_n\|^2 + \|M_{n+1}\|^2\right) \end{aligned} \tag{3.18}$$

Rearranging the inequality and using the stability of x_n, we arrive at

$$\begin{aligned} a(n)\|\nabla_x F(x_n)\|^2 \leq F(x_n) - F(x_{n+1}) - a(n)\langle \nabla_x F(x_n), e_n\rangle - a(n)\langle \nabla_x F(x_n), M_{n+1}\rangle \\ + a(n)^2 2LC \end{aligned} \tag{3.19}$$

for a sample path-dependent constant $C > 0$. Finally, sum the expressions over $n = 0, \ldots, t$ and evaluate the telescoping sum, then

$$\begin{aligned} \sum_{n=0}^{t} a(n)\|\nabla_x F(x_n)\|^2 \leq F(x_0) - F(x_{t+1}) - \sum_{n=0}^{t} a(n)\langle \nabla_x F(x_n), e_n\rangle \\ - \sum_{n=0}^{t} a(n)\langle \nabla_x F(x_n), M_{n+1}\rangle + \sum_{n=0}^{t} a(n)^2 2LC. \end{aligned} \tag{3.20}$$

The second summation on the right-hand side converges almost surely as $t \to \infty$ by the martingale convergence theorem (see A.2). Finally, use that $\sum_{n=0}^{\infty} a(n)^2 < \infty$, Cauchy-Schwarz and again the stability of x_n to conclude that

$$\sum_{n=0}^{t} a(n)\|\nabla_x F(x_n)\|^2 \leq C'\left(1 + \sum_{n=0}^{t} a(n)\|e_n\|\right) \tag{3.21}$$

for a sample path-dependent constants $C' > 0$. Thence,

$$\min_{n=0,\ldots,t} \|\nabla_x F(x_n)\|^2 \leq C'\left(\frac{1}{\sum_{n=0}^{t} a(n)} + \frac{\sum_{n=0}^{t} a(n)\|e_n\|}{\sum_{n=0}^{t} a(n)}\right) \tag{3.22}$$

The first term in Equation (3.22) is in line with the known almost sure rate of convergence estimates for SGD (Jun Liu and Yuan 2022, Theorem 1). The second term represents the reduction in the convergence rate due to AoI. As described in the paragraph prior to Theorem 3.5, the stability of x_n yields that $\|e_n\| \in \mathcal{O}\left(\sum_{m=n-\tau(n)}^{n-1} a(m)\right)$. Hence, (3.6) as stated in the theorem follows. □

Chapter 4

Distributed Asynchronous Reinforcement Learning

Multi-agent actor-critic algorithms (MAAC) are an important and popular class of Deep RL algorithms for intelligent decision-making in MAS (Arulkumaran et al. 2017). Popular MAAC algorithms, like the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm by Lowe et al. (2017), assume *instant access to global data (agent policies, their action sequences, and their global state information)* to train coordinated decentralized policies. This training paradigm is called *centralized training with decentralized execution*. This central perspective can be justified when the training involves an accurate simulator of an environment. In other scenarios, observations and decisions are inherently local and prone to communication errors and delays. Thus, algorithms that rely on a centralized paradigm are unsuitable *for truly distributed online multi-agent learning problems*. For such scenarios, we propose the Distributed Deep Deterministic Policy Gradient (3DPG) algorithm, a flexible, fully distributed algorithm for Markov Games that only uses local, potentially old communicated information. The 3DPG algorithm and its asymptotic convergence analysis were presented in (Redder, Ramaswamy, and Karl 2022a). Preliminary analysis and experiments were presented in (Redder, Ramaswamy, and Karl 2022c) and (Redder, Ramaswamy, and Karl 2022d), respectively.

3DPG is a naturally distributed extension of the single agent DDPG algorithm from Lillicrap et al. (2016). It enables coordinated but fully distributed online multi-agent learning. 3DPG learns coordinated policies using global data agents collected during training communicated over a network. Agents then use local data to make decisions. In other words, individual actions are not globally coordinated, but the average behavior is coordinated. 3DPG only requires that the agents exchange information (local states, actions, and policies) imperfectly: information might be lost or significantly delayed, causing data to have a random age once it arrives, which is described herein by AoI random variables. We analyze 3DPGs convergence under particularly

weak AoI assumptions for the MAS. As in the previous two chapters, the modest assumptions allow for potentially unbounded AoI with an arbitrary moment bound and make no deterministic requirements regarding data availability. Besides modest AoI assumptions, we require some additional conditions to ensure convergence.

Despite their popularity and usefulness in many practical scenarios, the conditions under which AC and MAAC algorithms converge are not well studied – we address this gap and discuss practically verifiable and sufficient conditions for DDPG, 3DPG, and MADDPG to converge asymptotically. The result is based on recent progress in understanding the convergence behavior of Deep Q-Learning (Ramaswamy and Hullermeier 2021). Such convergence guarantees and analyses are generally difficult, even for traditional *single-agent* DeepRL algorithms. In the single-agent case, RL algorithms with *linear* function approximations are well-studied, but algorithms that use *non-linear* function approximators like DNNs are poorly understood. At best, convergence is only characterized under strict assumptions that are difficult to verify in practice, e.g., L. Wang et al. (2019) assume that the state transition kernel of the Markov decision process is regular; this questions the practical usefulness of such analysis. The behavior of *multi-agent* DeepRL algorithms is even more challenging since the various agents' training processes are intertwined. Analyses with linear function approximation are therefore common (K. Zhang et al. 2018; Kumar, Koppel, and Ribeiro 2019). It is thus pertinent, both from a theoretical and a practical standpoint, to analyze, under practical assumptions, the asymptotic properties of multi-agent DeepRL algorithms with non-linear function approximation. In addition to these challenges, the analysis of 3DPG takes into account that agents communicate local data, resulting in DataAoI as introduced in Chapter 1. The considered modest AoI assumption determines how old the DataAoI can be, such that the 3DPG agents can converge jointly to a global policy of the state Markov process of the environment, which we define as a Markov game.

4.1 Markov games

In MAS, the global state of the environment is typically the concatenation of the agents' local states, while the global state is usually unobservable by any agent. The global state transitions to a new state after each agent has taken its local action. After the global state transition, the agents receive *local reward signals*. The structure of the local reward signals depends on the nature of the interaction between the agents: do they cooperate or compete?

As before, we use superscripts i for the agent index in the MAS. We assume that the MAS under consideration can be modeled as a D agent Markov game (Littman 1994), which is a decentralized Markov decision process with factored state-action spaces and agent-specific rewards. The high-level interaction of the D agents with their environment was already sketched in Figure 1.6.

Formally, a Markov game is defined as the 4-tuple $\left(\mathcal{S}, \mathcal{A}, p, \{r^i \mid 1 \leq i \leq D\}\right)$, with the following components.

- $\mathcal{S} = \prod_{i=1}^{D} \mathcal{S}^i$ is the global state space, with $\mathcal{S}^i \coloneqq \mathbb{R}^{k^i}$ the local state space of agent i.
- $\mathcal{A} = \prod_{i=1}^{D} \mathcal{A}^i$ is the global action space, where an action $a = (a^1, \ldots, a^D) \in \mathcal{A}$ denotes the joint action as a concatenation of local actions $a^i \in \mathcal{A}^i \subseteq \mathbb{R}^{d^i}, d^i > 0$.
- p is the Markov transition kernel, i.e. $p(\cdot \mid s, a)$ is the distribution of the successor state of state s after action a is executed.
- r^i is the local scalar reward function associated with agent i. Specifically, $r^i(s, a)$ is the local reward that agent i observes when the system is in state s and the global action a is taken.

In many cooperative Markov games, the local reward functions coincide, i.e., $r^i \equiv r$ for $1 \leq i \leq D$. Such models are called factored decentralized MDPs (Bernstein et al. 2002). We consider the more general Markov game model. As a consequence, the analysis covers a wide variety of scenarios, including cooperative and competitive ones.

Consider a D agent Markov game as defined in above. The D agents interact with the environment at discrete time steps $n \geq 0$. At every time step n, agent i observes a local state $s_n^i \in \mathcal{S}^i$, based upon which it must take a local (continuous) action $a_n^i \in \mathcal{A}^i \subseteq \mathbb{R}^{d^i}$, for which it receives a reward r_n^i. Suppose that the local behavior of agent i is defined by a local deep neural network (DNN) policy $\pi^i(s^i; \phi^i)$, parameterized by a vector ϕ^i. Define the associated global policy as

$$\pi \coloneqq (\pi^1, \ldots, \pi^D). \tag{4.1}$$

For each r^i, the return starting from time step 1 is defined by $R^i \coloneqq \sum_{n=1}^{\infty} \gamma^{n-1} r^i(s_n, a_n)$ with discount factor $0 < \gamma < 1$. Given a global policy π, the associated action-value function Q^i of agent i is given by $Q^i(s, a) \coloneqq \mathbb{E}_\pi \left[R^i \mid s_1 = s, a_1 = a\right]$. For each local reward function r^i, the associated optimal policy is characterized by the optimal action-value function $Q_*^i(s, a)$, which is defined as a solution to Bellman's equation (D. P. Bertsekas and J. N. Tsitsiklis 1996, Section 5.6):

$$Q_*^i(s, a) = \mathbb{E}_{s', r^i} \left[r^i(s, a) + \gamma \max_{a' \in \mathcal{A}} Q_*^i(s', a') \mid s, a \right] \tag{4.2}$$

The problem is to find local policy parametrizations ϕ_*^i for each local policy $\pi^i(s^i; \phi^i)$, such that for every agent i:

$$\pi^i(s^i; \phi_*^i) \approx \operatorname*{argmax}_{a^i \in \mathcal{A}} Q_*^i(s, a)\big|_{\pi^{j \neq i}} \quad \forall s \in \mathcal{S}. \tag{4.3}$$

In other words, all local policies should act optimally conditioned on the local policies obtained by all other agents. To achieve this, we propose 3DPG, a fully decentralized actor-critic algorithm.

4.2 The 3DPG algorithm

To get a first idea, let us view 3DPG as a multi-agent version of DDPG (Lillicrap et al. 2016), the most popular AC DeepRL algorithm. DDPG involves two DNNs: an actor (policy) network and a critic (Q-function) network. The actor-network is trained to approximate the optimal policy, and the critic network is trained to approximate the optimal Q-function. More specifically, the critic network is trained to minimize a variant of the squared Bellman error, while the actor-network is trained to pick actions that maximize the approximation of the optimal Q-function, as found by the critic. Notably, both the critic and actor networks are trained simultaneously.

In 3DPG, each agent only has access to a locally observable state (a part of the global state), can exchange information with other agents imperfectly, and takes actions that affect both its local state and states observable by other agents as described by the Markov game. To pick actions, each agent uses a local actor/policy. To train its local policy, each agent uses a local critic/Q-function approximation. The local policies are functions of the local agent states, constituting the global (multi-agent) system state. We assume that all agent clocks are discrete and synchronized. After all agents pick an action, they obtain local rewards at every discrete time step. Each agent then uses a local copy of DDPG (as explained above) to train a local actor while simultaneously training a local critic with respect to the global decision-making problem associated with its local reward structure.

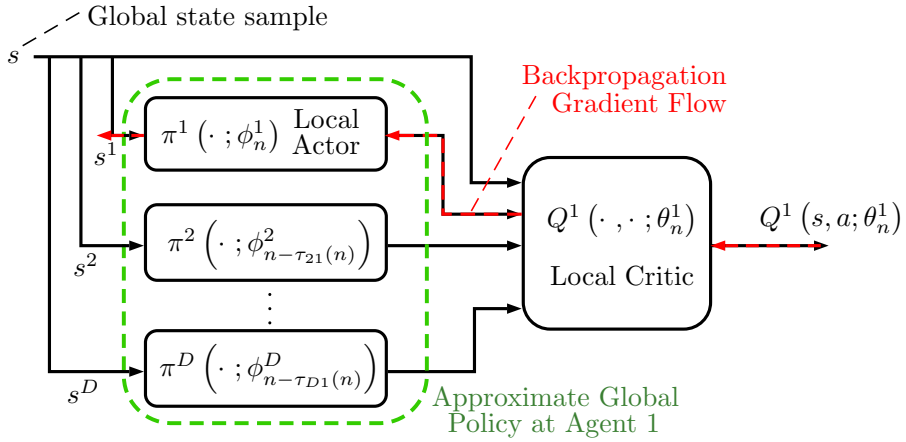

Figure 4.1: Illustration of the 3DPG architecture at agent 1. The local critic is evaluated for action $a = \pi(s;\phi_{\tau^1(n)})$ of the local approximation of the global policy. See Section 4.2.1 for details.

To perform the actor training step, the agents use local *policies* from other agents transferred via a potentially imperfect communication network that, e.g., causes delays. The training step at each agent may be viewed as calculating a local policy gradient using the best approximation of the current global policy. The resulting 3DPG architecture at agent 1 of a D agent system is illustrated in Figure 4.1. In addition to the old policies of other agents, all local actor and

critic training steps use data (states, actions) of other agents transferred via the communication network. This gives a quasi-centralized view based on information with potentially significant AoI to facilitate online training and execution of 3DPG.

Similar to the distributed SA iterations discussed in Chapter 2, we will conclude with results of Chapter 5 that 3DPG converges even when the AoI associated with the used information of other agents during training merely has an arbitrary moment bound. The convergence in the presence of such potentially significantly outdated information (the local states, actions, and policies from other agents) is achieved by using diminishing step-size sequences, ensuring that after some time, the change of the agents' local policies does not grow significantly larger than the step sizes. The asymptotic properties of 3DPG are:

1) All local critics converge to a set of DNN weights such that the local Bellman error gradients are zero on average with respect to limiting distributions over the global state-action space.
2) All local actors converge to a set of DNN weights such that the local policy gradients (taking into account the final policies of all other agents) are also zero with respect to the same limiting distributions.

The aforementioned limiting distributions are shaped by the agents' accumulated local experiences. More specifically, the global data tuples received by agents through communication form local limiting distributions such that all critics and actors converge jointly in expectation to stationary points with respect to these limiting distributions over the global state-action space. In that regard, no agent can improve its performance locally with respect to its limited distribution of the training process and the limited policies of the other agents. Under suitable assumptions, this implies that 3DPG agents converge to a local Nash equilibrium of Markov games. Notably, this is achieved although every agent may have a different local reward structure, i.e., irrespective of whether they cooperate or compete.

4.2.1 Algorithm description

Multi-agent actor-critic learning

To implement 3DPG for a Markov game, each agent i initializes a local DNN approximator $Q^i(s,a;\theta^i)$ for its local critic, where θ^i represents the associated vector of network weights. The local critic is trained using the Deep Q-Learning algorithm, (Mnih, Kavukcuoglu, Silver, et al. 2015), to find θ^i_* such that $Q^i(s,a;\theta^i_*) \approx Q^i_*(s,a)$ for all state-action pairs (s,a). Each agent further initilizes a local policy $\pi^i(s^i;\phi^i)$ parameterized by ϕ^i. The goal is to jointly train the local critics and actors so that (4.3) holds. This is challenging due to the potential conflicting rewards of the agents.

Consider that at some time step n, the local critic and actor parametrizations are θ^i_n and ϕ^i_n.

Further, suppose agent i gets access to a global data tuple

$$t_m^i := (s_m, a_m, r^i(s_m, a_m), s_{m+1}) \tag{4.4}$$

from some transition from time m to $m+1$ with $m \ll n$. The availability of at least some global tuples must be ensured by coordinated communication between the agents. This will be discussed further in the following subsections. Now, if agent i had access to the parametrizations of the other agents, then it could calculate a local critic gradient

$$\begin{aligned} &\nabla_{\theta^i} l^i(\theta_n^i, \phi_n, t_m^i) \\ &:= \nabla_{\theta^i} Q^i(s_m, a_m; \theta_n^i)\Big(r^i(s_m, a_m) + \gamma Q^i(s_{m+1}, \pi(s_{m+1}; \phi_n); \theta_n^i) - Q^i(s_m, a_m; \theta_n^i)\Big). \end{aligned} \tag{4.5}$$

Equation (4.5) is a sample gradient of the local squared Bellman error of Q^i for the observed global tuple t_m, which follows from the associated error in Bellmans equation (4.2).

Now, there are two possible ways to formulate a "natural" distributed version of the policy gradient in the DDPG algorithm The first one is the local policy gradient

$$\nabla_{\phi^i} g^i_{\text{MADDPG}}(\theta_n^i, \phi_n^i, s_m, a_m^{j\neq i}) := \nabla_{\phi^i}\pi^i(s_m^i; \phi_n^i)\nabla_{a^i} Q^i(s_m, a_m^1, \dots, \pi^i(s_m^i; \phi_n^i), \dots, a_m^D; \theta_n^i), \tag{4.6}$$

which is used in the MADDPG algorithm presented by (Lowe et al. 2017). The second one is

$$\nabla_{\phi^i} g^i_{\text{3DPG}}(\theta_n^i, \phi_n, s_m) := \nabla_{\phi^i}\pi^i(s_m^i; \phi_n^i)\nabla_{a^i} Q^i(s_m, \pi(s_m; \phi_n)); \theta_n^i), \tag{4.7}$$

which will be used in the 3DPG algorithm. The MADDPG local policy gradient uses the actions $a_m^{j\neq i}$ from the other agents from the global tuple t_m^i, while the local policy gradients in 3DPG use the policies $\phi_n^{j\neq i}$ from the other agents. *In the following, we use $\nabla_{\phi^i} g^i$ for (4.7) to simplify the notation.*

Remark 4.2.1. *The subtle difference between (4.6) and (4.7) will be analyzed in Section 4.4.3. Based on the asymptotic convergence analysis of MADDPG and 3DPG we will argue that a multi-agent actor-critic algorithm based on (4.7) has a higher probability of obtaining a better policy faster compared to a multi-agent actor-critic algorithm based on (4.6). This is because (4.7) takes precisely into account how other agents would behave in certain sampled states. Equation (4.6), on the other hand, also considers the sampled actions that may arise from randomly explored actions. The numerical experiments in Chapter 9 support this theoretical prediction.*

Remark 4.2.2. *The local policy gradient (4.7) seems to have a more direct motivation from the deterministic policy gradient theorem (DPGT) (Silver et al. 2014) in comparison to (4.6). The DDPG algorithm was inspired by taking samples from the DPGT. Similarly, the gradient (4.7) can be motivated by inserting the global product policy (4.1) into the DPGT. Thus Equation (4.7) is, in essence, the policy gradient from the DDPG algorithm (Lillicrap et al. 2016), where the*

policy is defined in product form (4.1). *Finally, it must be noted that the sample gradients used in DDPG, MADDPG, and 3DPG are not true sample gradients from the deterministic policy gradient (Nota and Thomas 2019).*

The idea behind 3DPG is to approximate (4.5) and (4.7) using old information from other agents to train Q^i and π^i locally. To implement this, the agents require:

1) Local access to global data tuples t_m^i,
2) Local access to the global policy $\pi(s;\phi_m)$,

for $m \ll n$ "frequently" (the precise network assumptions are presented in section 4.2.2). Recall that in MADDPG, the above information is required for all agents at every time step. These are reasonable assumptions for simulated environments or under the paradigm of centralized training with decentralized execution but not for online, fully distributed learning.

Approximate global policy induced by local AoI

Using communication, we now decentralize the implementation of (4.5) and (4.7); we use communicated but potentially aged local policies to approximate the true global policy ϕ_n. We suggest a communication paradigm where agents cooperatively forward local data to other agents, such that local policies and local data (states and actions) can flow via the network to all other agents. To guarantee this, the agents must use a flooding protocol, e.g., (Lim and Kim 2001), to forward old policies ϕ_n^i between the agents as well as a coordinated communication protocol to ensure that at least some global tuples (4.4) reach each agent "frequently". For the coordinated communication protocol, one could use broadcast protocol coupled with a central coordinator or a distributed snapshot protocol (Chandy and Lamport 1985), which would cost more communication resources. For now, suppose that the agents run suitable protocols of this kind.

Consider now that each agent runs a local algorithm to train its policy, generating a sequence ϕ_n^i of associated policy parametrizations. Equipped with the ability to transfer data via the available network, the agents exchange the local parametrization ϕ_n^i as well as local tuples $t_n^i \coloneqq (s_n^i, a_n^i, s_{n+1}^i)$ using the communication network that possibly delays or loses data for extended periods. Hence, agent i has only access to $\phi^j_{n-\tau_{ij}(n)}$ for every agent $j \neq i$ at every time step n. Here, $\phi^j_{n-\tau_{ij}(n)}$ denotes the latest available policy parametrization from agent j at agent i at time n and, as in the previous section, we refer to $\tau_{ij}(n)$ as the associated AoI process as a consequence of the potentially uncertain and delaying communication. For every agent i, we can then define a global policy parametrization associated with the aged information at time n by

$$\phi_{\tau^i(n)} \coloneqq (\phi^1_{n-\tau_{i1}(n)}, \dots, \phi^D_{n-\tau_{D1}(n)}). \tag{4.8}$$

This global policy will serve as an approximation to the true global policy ϕ_n.

The 3DPG algorithm

The previous paragraph explained how a D agent multi-agent system has to use an available network to exchange their local policy parametrizations ϕ_n^i and their local tuples $(s_n^i, a_n^i, s_{n+1}^i)$. We can now state the 3DPG iterations.

Suppose that every agent i maintains a local replay memory $\mathcal{R}_n^i$. At every time step n, the memory can contain up to N old global transitions t_m^i. At time step n, agent i samples a random minibatch of $M < N$ transitions from its replay memory. Agent i then updates its critic and critic using step-size sequences $\alpha(n)$ and $\beta(n)$, respectively:

$$\begin{aligned}\theta_{n+1}^i &= \theta_n^i + \alpha(n)\frac{1}{M}\sum_m \nabla_{\theta^i} l^i(\theta_n^i, \phi_{\tau^i(n)}, t_m^i)\\ \phi_{n+1}^i &= \phi_n^i + \beta(n)\frac{1}{M}\sum_m \nabla_{\phi^i} g^i(\theta_m^i, \phi_{\tau^i(n)}, s_m)\end{aligned} \tag{4.9}$$

Notice that for a single sample, the gradients used in (4.9) are the gradients (4.5) and (4.7) where the global policy ϕ_n has been replaced by the local approximation of the global policy $\phi_{\tau^i(n)}$ induced by the aged parametrization (4.8). The resulting training architecture was already presented in Figure 4.1; the figure shows the backpropagated gradient flow for 3DPG during training. The detailed training algorithm will be presented in Algorithm 5 alongside the numerical experiments in Chapter 9. We will now present the assumptions to prove the convergence of MAAC iteration (5). We begin with the required network/AoI assumptions.

4.2.2 Assumptions

Network assumptions

The communication network needs to ensure two things. First, it must ensure that every agent i receives the policy parametrizations ϕ_n^j for all $j \neq i$ "sufficiently" often. Second, it needs to ensure that the available samples in the replay memories $\mathcal{R}_n^i$ are not too old. To capture the age of the samples in the replay memories, define the DataAoI random variable $\Delta^i(n)$ as the age of the oldest sample in the replay memory $\mathcal{R}_n^i$ of agent i at time $n \geq 0$. With the DataAoI random variables, the 3DPG iterations (4.9) can now be expressed in the form of the introductory iteration (1.2), with AoI $\tau_{ij}(n)$ and $\Delta^i(n)$.

Assumption 4.2.1.

(a) Policy communication assumptions:
For all i, j, the actor-critic stepsize $\alpha(n)$, $\beta(n)$ and the AoI sequences $\tau_{ij}(n)$ guarantee that $\sum_{k=n-\tau_{ij}(n)}^{n-1} \alpha(k) \to 0$ a.s. and $\sum_{k=n-\tau_{ij}(n)}^{n-1} \beta(k) \to 0$ a.s.

(b) For all i, j, the actor-critic stepsize $\alpha(n)$, $\beta(n)$ and the AoI sequences $\tau_{ij}(n)$ guarantee that $\sum_{k=n-\Delta^i(n)}^{n-1} \alpha(k) \to 0$ a.s. and $\sum_{k=n-\Delta^i(n)}^{n-1} \beta(k) \to 0$ a.s.

As before, Assumption 4.2.1 requires a trade-off between the tail distribution decay of the AoI variables $\tau_{ij}(n)$ and $\Delta^i(n)$ and the choice of the AC stepsize sequences. We will see in the next chapter that this can be ensured with a dominating random variable with an arbitrary moment. As in Chapter 2, Assumption 4.2.1 ensures that the use of the approximate global policies $\phi_{\tau^i(n)}$ instead of the true global policy ϕ_n in (4.9) does not cause gradient errors asymptotically. Finally, Assumption 4.2.1(b) is used in Lemma 4.9 to show that the agents' experiences converge to a stationary distribution. Specifically, Assumption 4.2.1(b) ensures that the agents receive enough global tuples asymptotically to "track" the Markov game state distribution, which will answer (Q3) for the presented MARL setting. This is the first time that these assumptions are considered for data availability in multi-agent learning.

Remark 4.2.3. *Assumption 4.2.1(b) does not specify when exactly the global samples become available to each agent and the received data tuples do not have to be from the same time steps m for every agent.*

Algorithm and Markov game assumptions

In addition to the network assumptions, we require the following assumptions:

Assumption 4.2.2.

(a) The critic step size sequence $\alpha(n)$ is positive, monotonically decreasing and satisfies: $\alpha(n) \in \mathcal{O}\left(n^{-q}\right)$ for $q \in (\frac{1}{2}, 1]$.

(b) The actor step size sequence $\beta(n)$ satisfies:
$\sum_{n\geq 0}\left(\alpha(n)-\beta(n)\right) < \infty$

Assumption 4.2.3.

(a) $\sup_{n\geq 0}\|\theta_n\| < \infty$ a.s. and $\sup_{n\geq 0}\|\phi_n\| < \infty$ a.s.

(b) $\sup_{n\geq 0}\|s_n\| < \infty$ a.s. and the action space $\mathcal{A}$ is compact.

Assumption 4.2.4. *The state transition kernel $p(\cdot \mid s, a)$ is continuous.*

Assumption 4.2.5. *The actor policies π^i, and the critics Q^i are fully connected feedforward neural networks with twice continuously differentiable activation functions such as Gaussian Error Linear Units (GELUs).*

Assumption 4.2.6. *The reward functions $r^i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ are continuous.*

Assumption Assumption 4.2.3(a) is the strongest assumption as it requires almost sure stability of the algorithm. Practically, we are, therefore, characterizing 3DPG limit points if 3DPG is observed to be stable. We will further discuss the stability assumption in the chapter-specific conclusions and related work Section 4.5. However, the results can also be combined with recent

results on clipped gradient-based algorithms (Ramaswamy, Shalabh Bhatnagar, and Saxena 2023), which will enable us to remove the algorithm stability assumption.

The compactness of the action space in Assumption 4.2.3(b) will usually be satisfied in many applications, e.g., in robotics. The Assumption 4.2.3(a) can be relaxed to an arbitrary real dimensional space without a sample path-dependent boundedness condition; see (Ramaswamy and Hullermeier 2021).

In Assumption 4.2.1(b), we require that the critic and actor step-size sequences are chosen such the critics and actors asymptotically run on the same time scale, i.e., updated in the same order of magnitude. An example step size sequence is presented in Figure 4.2. A necessary but not

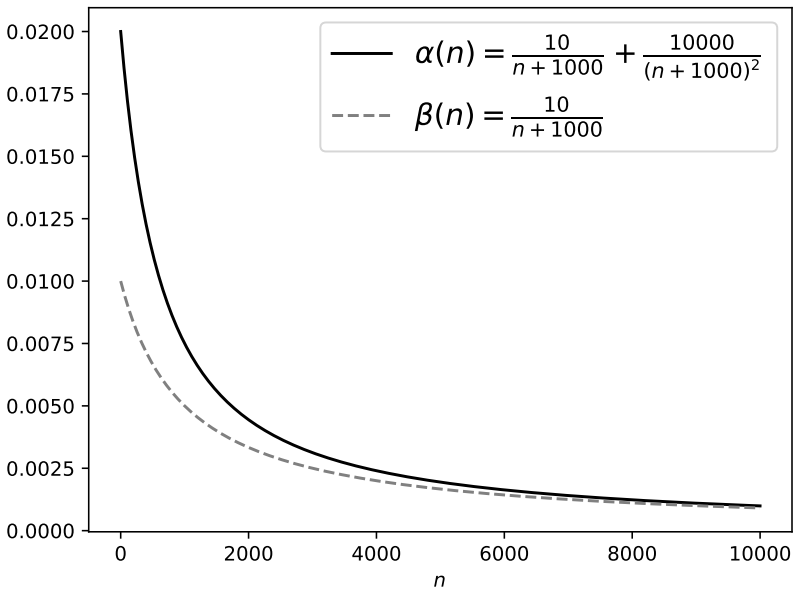

Figure 4.2: Illustration of step size sequences that satisfy Assumption 4.2.1

sufficient condition for Assumption 4.2.1(b) is that $\frac{\beta(n)}{\alpha(n)} \to 1$. This is not a traditional assumption for actor-critc algorithms (V. S. Borkar and Konda 1997). The traditional actor-critic analysis assumes $\frac{\beta(n)}{\alpha(n)} \to 0$, which implies that the critic runs on a faster time scale. Instead, we will present a proof based on a single-timescale analysis of (4.9) with respect to the timescale of the critic iterations. In practice, we may still want the critic to converge faster, so we would initially choose $\alpha(n)$ larger than $\beta(n)$. Assumption 4.2.1(b) requires that afterward, the iterations asymptotically take steps of asymptotically the same size. The more difficult analysis using a two-timescale step-size schedule is still an open question, which requires the study of set-valued dynamics.

Remark 4.2.4. *It should be noted that although a two-timescale analysis with a critic on a faster time-scale has a long tradition and an intuitive motivation, a convergence analysis of critic-actor*

learning for tabular settings has also presently been proposed by Shalabh Bhatnagar, V. S. Borkar, and Guin (2023). Here, the actor is the one running on the faster time-scale. A single time-scale analysis, as discussed herein, can thus be seen as a merge of the aforementioned two perspectives.

In Assumption 4.2.5, we require twice continuous differentiable activations used by the policy and actor networks. GELUs are well-known examples that satisfy this property (Hendrycks and Gimpel 2016). Additionally, GELUs are a well-known neural network activation function with similar high performance across different tasks compared to other well-known activations like ELUs or LeakyReLUs (Ramachandran, Zoph, and Le 2017). We now present the convergence theorem.

4.3 Convergence theorem and Nash equilibria

Theorem 4.1. *Under Assumption 4.2.1-4.2.6, the 3DPG iterations (4.9) converge to limits θ^i_∞ and ϕ^i_∞ almost surely, such that*

$$\nabla_{\theta^i}\left(\int_{\mathcal{S}\times\mathcal{A}} l^i(\theta^i_\infty,\phi_\infty,s,a)\mu^i_\infty(ds,da)\right)=0, \tag{4.10}$$

$$\nabla_{\phi^i}\left(\int_{\mathcal{S}} g^i(\theta^i_\infty,\phi_\infty,s)\mu^i_\infty(ds,\mathcal{A})\right)=0, \tag{4.11}$$

where μ^i_∞ is a limiting distribution of a continuous time measure process (defined in Section 4.4.2) that captures the experience of agent i sampled from its local experience replay $\mathcal{R}^i_n$ during training. Further, all μ^i_∞ are stationary distributions of the state Markov process:

$$\mu^i_\infty(dy\times\mathcal{A})=\int_{\mathcal{S}} p(dy\mid x,\pi(x;\phi_\infty))\mu^i_\infty(dx\times\mathcal{A}). \tag{4.12}$$

Theorem 4.1 shows that the critic iterations of 3DPG converge to stationary points of the average local squared Bellmann errors. Further, the actor iterations converge to stationary points of the average local deterministic policy gradients. For both limits, the averaging is with respect to the stationary distributions of the state Markov process that captures the experienced samples of the agents.

The next vital question is whether the stationary points guaranteed by Theorem 4.1 are local minima for the critics and local maxima for the actors, but not saddle points. This is generally a challenging problem in non-convex optimization, but we shall discuss briefly how to tackle it. First, since stochastic gradient descent schemes tend to avoid unstable equilibria, (Mertikopoulos et al. 2020; Vlaski and Sayed 2021; Ge et al. 2015), we can expect that the aforementioned stationary points are local minima/maxima with high probability. We plan to make this more precise in the future using avoidance of traps analysis (V. Borkar 2022, Ch. 4), which requires assumptions on the noise effecting the 3DPG iteration to ensure that unstable equilibria are

avoited. With this, we could conclude that 3DPG converges to the local solution of the objective (4.3) with high probability.

The previous paragraph highlights that we can generally expect 3DPG to converge to local minima under suitable noise conditions. In this case, it is immediately from the policy gradient update of the actors that the limit is a local approximate Nash equilibrium. To formulate such a statement, denote the final local policies by $\pi^i_\infty(s^i) \coloneqq \pi^i(s^i;\phi^i_\infty)$. For any open set $\mathcal{U}$ with $0 \in \mathcal{U}$ in the parameter space of $\pi^i_\infty(s^i)$, define the open neighborhood $\Pi^i_\infty(\mathcal{U}) \coloneqq \{\pi^i(\,\cdot\,;\phi^i) : \phi^i \in \phi^i_\infty + \mathcal{U}\}$ in the space of policies.

Corollary 4.2. *Suppose that the stationary point of the local policies are local maxima, then there are open sets $\mathcal{U}^i$ with $0 \in \mathcal{U}^i$, such that*

$$\begin{aligned}\int_{\mathcal{S}} Q^i(s,\pi^1_\infty(s),\dots,\pi^i_\infty(s),\dots,\pi^D_\infty(s);\theta^i_\infty)\mu^i_\infty(s,\mathcal{A}) \\ \geq \int_{\mathcal{S}} Q^i(s,\pi^1_\infty(s),\dots,\pi^i(s),\dots,\pi^D_\infty(s);\theta^i_\infty)\mu^i_\infty(s,\mathcal{A})\end{aligned} \tag{4.13}$$

for all $\pi^i \in \Pi^i_\infty(\mathcal{U}^i)$.

Proof. Follows by assumption, as $\pi^i_\infty(s)$ would be a local maximum of

$$\int_{\mathcal{S}} Q^i(s,\pi^1_\infty(s),\dots,\pi,\cdot,\dots,\pi^D_\infty(s);\theta^i_\infty)\mu^i_\infty(s,\mathcal{A})$$

in the space of policies π with the same DNN model as $\pi^i_\infty(s)$. □

The statement of Corollary 4.2 describes that for a given local reward structure, the agents converge to an equilibrium where they have no desire to change their policies by a small amount given their local experience and the final policies of the other agents. In other words, (4.13) formalized that agents converge to a local Nash equilibrium, (Prasad, LA, and Shalabh Bhatnagar 2015), with respect to their locally approximated action-value functions. Specifically, the local policies converge to a local Nash equilibrium with respect to the local expected action-value functions $\int_{\mathcal{S}} Q^i(s,\cdot\,;\theta^i_\infty)\mu^i_\infty(s,\mathcal{A})$ based on the experience gathered by each agent represented by the local limiting distributions μ^i_∞. In game theoretic terms, these are the payoff (or utility) functions with respect to which the agents converge to a local Nash equilibrium.

The significance of corollary 4.2 is that the agents converge to a local approximate Nash equilibrium without assuming that the samples used in training are from a known stationary distribution of the state Markov process. Instead, it is shown that the experience of the agents gives rise to stationary distributions of the state Markov process. This is important, as deep MARL algorithms are typically employed in complex environments with multiple stationary distributions.

Finally, we close this section with a promising set of assumptions that can guarantee convergence to a global approximate Nash equilibrium.

Assumption 4.3.1. *Assume that*

1) *the local policies are parameterized by a linear combination of non-linear features.*
2) *the local critics are parameterized, such that each $-Q^i(s,a;\theta^i)$ is an input-convex neural network in the i-th action input (Amos, L. Xu, and Kolter 2017) with softplus activation function (Goodfellow, Bengio, and Courville 2016).*

Corollary 4.3. *Suppose that Assumption 4.3.1 holds and that the limiting critic weights are non-negative (i.e., the coordinates of θ^i are non-negative), then*

$$\int_{\mathcal{S}} Q^i(s,\pi^1_\infty(s),\dots,\pi^i_\infty(s),\dots,\pi^D_\infty(s);\theta^i_\infty)\mu^i_\infty(s,\mathcal{A}) \tag{4.14}$$
$$\geq \int_{\mathcal{S}} Q^i(s,\pi^1_\infty(s),\dots,\pi^i(s),\dots,\pi^D_\infty(s);\theta^i_\infty)\mu^i_\infty(s,\mathcal{A})$$

for all policies π^i with the same parameter space as π^i_∞

Proof. Under Assumption 4.3.1, Amos, L. Xu, and Kolter (2017, Prop. 1) show that $-Q^i(s,a;\theta^i)$ is input convex in the i-th action coordinate since $-Q^i(s,a;\theta^i)$ has strictly monotone activations and the limiting weights are by assumption non-negative. Combined with the linearity of the policy, it follows that $-Q^i(s,\pi^1_\infty(s),\dots,\pi^i(s^i;\phi^i_\infty),\dots,\pi^D_\infty(s);\theta^i_\infty)$ is a convex function in ϕ^i_∞. As expectations preserve convexity, ϕ^i_∞ is guaranteed to be the global maximum of $\int_{\mathcal{S}} Q^i(s,\pi^1_\infty(s),\dots,\pi^i(s),\dots,\pi^D_\infty(s);\theta^i_\infty)\mu^i_\infty(s,\mathcal{A})$ over the space of space of linear policies parameterized by the given set of non-linear features. This concludes the proof. □

Corollary 4.3 presents a verifiable set of assumptions to ensure that 3DPG policies converge to global approximate Nash equilibrium w.r.t. to approximated local critics. The only required assumption aside from implementing the architecture from Amos, L. Xu, and Kolter (2017) with softplus activation function is that the final neural network weights are non-negative. This can be ensured with high probability using sufficient weight regularization. However, in the future, we plan to ensure this property systematically with a constraint optimization approach compatible with the analysis to be presented in the next sections.

4.4 Analysis

4.4.1 Preliminaries and Age-of-Information analysis

In the following two sections, we prove Theorem 4.1. All technical proofs are presented in Appendix 4.6. The proof builds on the analysis of single-agent deep Q-learning presented in (Ramaswamy and Hullermeier 2021). Several changes are necessary to move to the online MAAC learning setting considered herein. To simplify the presentation, we will assume that the state

space $\mathcal{S}$ is compact. All results can be generalized to d-dimensional real spaces under the almost sure boundedness condition in Assumption 4.2.3(b), for which we refer to techniques presented in (Ramaswamy and Hullermeier 2021, Section IV.A.2). After we reduce 3DPG to an iteration without AoI, we will present at its core a convergence proof for the DDPG algorithm (Lillicrap et al. 2016), using a single timescale analysis. We now begin with preliminary reductions and the analysis of the AoI processes $\tau_{ij}(n)$.

Reduction to mini-batches of size 1

First, we make a simplifying reduction. We consider that the agents have ready access to the global tuples $(s_n, a_n, r^i(s_n, a_n), s_{n+1})$ during runtime and that merely the local policies ϕ_n^i are communicated via the communication network. Further, we merely consider that the agents use the global tuple from time n to update its critic and actor network. We therefore simplify iteration (4.9) to:

$$\begin{aligned} \theta_{n+1}^i &= \theta_n^i + \alpha(n)\nabla_{\theta^i} l(\theta_n^i, \phi_{\tau^i(n)}, t_n^i), \\ \phi_{n+1}^i &= \phi_n + \beta(n)\nabla_{\phi^i} g(\theta_n^i, \phi_{\tau^i(n)}, s_n). \end{aligned} \tag{4.15}$$

In Section 4.4.2, we will extend the analysis to the full setting presented in Section 4.2.

Reduction to zero AoI

As the second step, we define the gradient errors that occur since we use the aged global policies $\phi_{\tau^i(n)}$ instead of the true global policy: $e_n^{\theta^i} := \nabla_{\theta^i} l(\theta_n^i, \phi_n, t_n) - \nabla_{\theta^i} l(\theta_n^i, \phi_{\tau^i(n)}, t_n)$, $e_n^{\phi^i} := \nabla_{\phi^i} g(\theta_n^i, \phi_n, s_n) - \nabla_{\phi^i} g(\theta_n^i, \phi_{\tau^i(n)}, s_n)$. Hence, (4.15) can be written as

$$\begin{aligned} \theta_{n+1}^i &= \theta_n^i + \alpha(n)\left(\nabla_{\theta^i} l(\theta_n^i, \phi_n, t_n) + e_n^{\theta^i}\right), \\ \phi_{n+1}^i &= \phi_n + \beta(n)\left(\nabla_{\phi^i} g(\theta_n^i, \phi_n, s_n) + e_n^{\phi^i}\right). \end{aligned} \tag{4.16}$$

Reduction to marginalized critic gradient

As the third step, we rewrite the critic iterations in (4.16) further by integrating over the successor state s_{n+1} in t_n given state s_n. The resulting new loss gradient is $\nabla_{\theta^i}\hat{l}^i(\theta^i, \phi, s, a) :=$

$$\left(r^i(s,a) + \gamma \int Q^i(s', \pi(s';\phi); \theta^i) p(ds' \mid s, \phi) - Q^i(s, a; \theta^i)\right) \nabla_{\theta^i} Q^i(s, a; \theta^i), \tag{4.17}$$

With a slight abuse of notation, we use $p(ds \mid s_n, \phi_n)$ instead of $p(ds \mid s_n, \phi_n(s_n))$ to highlight the dependency of the action a_n on the policy ϕ_n and potential additional random noise for exploration. Define the induced error from using $\nabla_{\theta^i}\hat{l}^i$ instead of $\nabla_{\theta^i} l^i$ as $\psi_n^i := \nabla_{\theta^i}\hat{l}^i - \nabla_{\theta^i} l^i$,

then

$$\begin{aligned}\theta^i_{n+1} &= \theta^i_n + \alpha(n)\left(\nabla_{\theta^i}\hat{l}^i(\theta^i_n, \phi_n, s_n, a_n) + \psi^i_n + e^{\theta^i}_n\right),\\ \phi^i_{n+1} &= \phi_n + \beta(n)\left(\nabla_{\phi^i} g(\theta^i_n, \phi_n, s_n) + e^{\phi^i}_n\right).\end{aligned} \tag{4.18}$$

$e^{\theta^i}_n$, $e^{\phi^i}_n$ and ψ^i_n vanish asymptotically

In summary, we have rewritten (4.15) using: 1. The errors $e^{\theta^i}_n$ and $e^{\phi^i}_n$ induced by not considering the AoI random variables $\tau_{ij}(n)$, and 2. The errors ψ^i_n induced by marginalizing out the successor states s_{n+1}. Using that the activation functions of the actors and critics are twice continuously differentiable, it now follows that the loss gradients $\nabla_{\phi^i} g^i$, $\nabla_{\theta^i} l^i$ and $\nabla_{\theta^i}\hat{l}^i$ are locally Lipschitz.

Lemma 4.4. *(i) $\nabla_{\theta^i} l^i(\theta^i_n, \phi_n, t_n)$ and $\nabla_{\theta^i}\hat{l}^i(\theta^i_n, \phi_n, s_n, a_n)$ are continuous and locally Lipschitz continuous in the θ^i and ϕ-coordinate.*
(ii) $\nabla_{\phi^i} g(\theta^i_n, \phi_n, s_n, a_n)$ is locally Lipschitz continuous in every coordinate.

We can now show that the errors $e^{\theta^i}_n$ and $e^{\phi^i}_n$ due to AoI vanish asymptotically.

Lemma 4.5. $\lim_{n\to\infty} \|e^{\theta^i}_n\| = 0$ *and* $\lim_{n\to\infty} \|e^{\phi^i}_n\| = 0$.

The next lemma shows that the accumulated errors due to the marginalization of the successor states in (4.18) are convergent almost surely. It therefore follows that ψ^i_n vanishes.

Lemma 4.6. $\Psi^i_n := \sum_{m=0}^{n-1} \alpha(m)\psi^i_n$ *is a zero-mean square integrable martingale. Hence, Ψ^i_n converges almost surely.*

It now follows from Lemma 4.5 and Lemma 4.6 that we can study the convergence of (4.18) without the additional error terms $e^{\theta^i}_n$, $e^{\phi^i}_n$ and ψ^i_n. This is because the error terms will contribute additional asymptotically negligible errors in the following proof of Lemma 4.7. With a slide abuse of notation, we now redefine the critic loss gradients $\nabla_{\theta^i} l^i$ as the marginalized critic loss gradient $\nabla_{\theta^i}\hat{l}^i$.

4.4.2 Convergence analysis

To analyze the asymptotic behavior of (4.18), we again follow the ODE approach from SA (V. Borkar 2022), i.e., we construct a continuous-time trajectory with the same limiting behavior as (4.18). First, we divide the time axis using $\alpha(n)$ as follows: $t_0 := 0$ and $t_n := \sum_{m=0}^{n-1} \alpha(m)$ for $n \geq 1$. Now define

$$\overline{\theta}^i(t_n) := \theta^i_n, n \geq 0 \text{ and } \overline{\phi}^i(t_n) := \phi^i_n, n \geq 0. \tag{4.19}$$

Let $\mathbb{R}^{p^i_\theta}$ and $\mathbb{R}^{p^i_\phi}$ be the parameter spaces of the θ^i_n's and ϕ^i_n's, respectively. Then define $\overline{\theta}^i \in \mathbb{C}([0,\infty), \mathbb{R}^{p^i_\theta})$ and $\overline{\phi}^i \in \mathbb{C}([0,\infty), \mathbb{R}^{p^i_\phi})$ by linear interpolation of all $\overline{\theta}^i(t_n)$ and $\overline{\phi}^i(t_n)$, respectively.

To analyze the training process, we formulate a measure process that captures the encountered state-action pairs when using the global policy $\pi(s_n; \phi_n)$. Therefore, define

$$\mu(t) = \delta_{(s_n, a_n)}, t \in [t_n, t_{n+1}] \tag{4.20}$$

where $\delta_{(x,a)}$ denotes the Dirac measure. This defines a process of probability measures on $\mathcal{S} \times \mathcal{A}$. For every probability measure ν on $\mathcal{S} \times \mathcal{A}$, define

$$\begin{aligned} \tilde{\nabla} l^i(\theta^i, \phi, \nu) &:= \int \nabla_{\theta^i} l^i(\theta^i, \phi, s, a) \nu(ds, da), \\ \tilde{\nabla} g^i(\theta^i, \phi, \nu) &:= \int \nabla_{\phi^i} g^i(\theta^i, \phi, s) \nu(ds, \mathcal{A}). \end{aligned} \tag{4.21}$$

Note that in $\nabla_{\phi^i} g^i$ we used $\nu(ds, \mathcal{A})$, since the actor update in (4.18) is only state-dependent. It follows from Lemma 4.4 that all $\tilde{\nabla} l^i$ and $\tilde{\nabla} g^i$ are continuous in all coordinates and locally Lipschitz in both the θ^i- and ϕ-coordinate.

We can now define the associated continuous time trajectories in $\mathbb{C}([0, \infty), \mathbb{R}^{p^i_\theta})$ and $\mathbb{C}([0, \infty), \mathbb{R}^{p^i_\phi})$ that capture the training process starting from time t_n for $n \geq 0$:

$$\begin{aligned} \theta^i_n(t) &:= \overline{\theta}^i(t_n) + \int_0^t \tilde{\nabla} l^i(\theta^i_n(x), \phi_n(x), \mu_n(x)) dx, \\ \phi^i_n(t) &:= \overline{\phi}^i(t_n) + \int_0^t \tilde{\nabla} g^i(\theta^i_n(x), \phi_n(x), \mu_n(x)) dx. \end{aligned} \tag{4.22}$$

where $\mu_n(t) := \mu(t_n + t)$. The combination of the continuous-time trajectories in (4.22) results in the aforementioned single trajectory with the same limiting behavior as (4.18). Per definition, the trajectories define solutions to the following families of non-autonomous ordinary differential equations (ODEs):

$$\begin{aligned} \{\dot{\theta}^i_n(t) &= \tilde{\nabla} l^i(\theta^i_n(t), \phi_n(t), \mu_n(t))\}_{n \geq 0}, \\ \{\dot{\phi}^i_n(t) &= \tilde{\nabla} g^i(\theta^i_n(t), \phi_n(t), \mu_n(t))\}_{n \geq 0}. \end{aligned} \tag{4.23}$$

By construction, we obtain that the limiting behavior of (4.18) is captured by the limits of the sequences $\{\overline{\theta}^i([t_n, \infty))\}_{n \geq 0}$ and $\{\overline{\phi}^i([t_n, \infty))\}_{n \geq 0}$ defined by (4.19). Further, the sequences defined in (4.22) can be analyzed as solutions to the ODEs in (4.23). If (4.19) and (4.22) behave asymptotically identical, then the limiting behavior of (4.18) is thus captured by the solutions to the ODEs in (4.23). This is formalized by the following important technical Lemma 4.7. This lemma is the key component that enables a single-timescale analysis of DDPG-style actor-critic algorithms. To prove Lemma 4.7, we use that the step size sequences are related by $\sum_{n \geq 0} (\alpha(n) - \beta(n)) < \infty$ from Assumption 4.2.1(b). This is essential since we just constructed the continuous trajectories with respect to the timescale induced by $\alpha(n)$. The assumption, in essence, requires that the critic and actor update asymptotically run on the same time scale.

Lemma 4.7. *For every $T > 0$, we have*

$$\lim_{n\to\infty} \sup_{t\in[0,T]} \|\overline{\theta}^i(t_n + t) - \theta_n^i(t)\| = 0, \tag{4.24}$$

$$\lim_{n\to\infty} \sup_{t\in[0,T]} \|\overline{\phi}^i(t_n + t) - \phi_n^i(t)\| = 0. \tag{4.25}$$

Lemma 4.7 shows that we can analyze the limits of (4.18) as the limits of the continuous-time trajectories defined in (4.19) in conjunction with the measure process (4.20). By construction, the trajectories $\theta_n^i(t)$ and $\phi_n^i(t)$ are equicontinuous. Moreover, they are point-wise bounded from Assumption 4.2.3(a). It now follows from the Arzela-Ascoli theorem, see A.1, that the families of trajectories

$$\{\theta_n^i([0,\infty))\}_{n=0}^{\infty}, \qquad \{\phi_n^i([0,\infty))\}_{n=0}^{\infty} \tag{4.26}$$

are sequentially compact in $\mathbb{C}([0,\infty), \mathbb{R}^{p_\theta^i})$ and $C([0,\infty), \mathbb{R}^{p_\phi^i})$, respectively. Further, it can be shown that the space of measurable functions from $[0,\infty)$ to the space of probability measures on $\mathcal{S}\times\mathcal{A}$ is compact metrizable (V. S. Borkar 2006). It now follows that the product space of all trajectories $\theta_n^i(t)$ and $\phi_n^i(t)$ together with the aforementioned space of measurable functions is sequentially compact. Hence, there is a common subsequence such that all considered sequences converge simultaneously, i.e. we obtain (with a slight abuse of notation) that $\theta_n^i \to \theta_\infty^i$ in $C([0,\infty), \mathbb{R}^{p_\theta^i})$, $\phi_n^i \to \phi_\infty^i$ in $C([0,\infty), \mathbb{R}^{p_\phi^i})$ and $\mu_n \to \mu_\infty$ in the space of measurable functions. Analogously to (Ramaswamy and Hullermeier 2021, Lemma 4), we can show that $\mu_n(t)$ also converges in distribution to $\mu_\infty(t)$ in $\mathbb{P}(\mathcal{S}\times\mathcal{A})$, t almost everywhere.

The following lemma now shows that the limits θ_∞^i, ϕ_∞^i and μ_∞ are solutions to the limits of the families of non-autonomous ordinary differential equations (4.23).

Lemma 4.8. *a) θ_∞^i is a solution to $\dot{\theta}^i(t) = \tilde{\nabla} l^i(\theta^i(t), \phi_\infty^1(t), \ldots, \phi_\infty^D(t), \mu_\infty(t))$*
b) ϕ_∞^i is a solution to $\dot{\phi}^i(t) = \tilde{\nabla} g^i(\theta_\infty^i(t), \phi_\infty^1(t), \ldots, \phi^i(t), \ldots, \phi_\infty^D(t), \mu_\infty(t))$

We can now study the limit of 3DPG as a solution to the aforementioned non-autonomous ODEs. Specifically, append the ODEs to form a new ODE in the appended parameter space. The rest of the analysis follows the line of argument in (Ramaswamy and Hullermeier 2021, Thm. 1), so we only state the main conclusion. Let $(\theta^1, \ldots, \theta^D, \phi^1, \ldots, \phi^D)$ be a solution to the appended ODE, then the solution converges to an equilibrium of the appended ODE, i.e. $\tilde{\nabla} l^i(\theta^i, \phi, \overline{\mu}_\infty^i) = 0$ and $\tilde{\nabla} g^i(\theta^i, \phi, \overline{\mu}_\infty^i) = 0$ for all i, where $\lim_{t\to\infty} \mu_\infty^i(t) \overset{d}{=} \overline{\mu}_\infty^i$. Lemma 4.7 and Lemma 4.8 show that the joint of the sequences $\overline{\theta}^i(t_{n(k)})$ and $\overline{\theta}^i(t_{n(k)})$ are solutions to the appended ODE for $\{n(k)\}_{k\geq 0} \subset \{n\}_{n\geq 0}$. The last two statements thus show that the limits of $\overline{\theta}^i(t_{n(k)})$ and $\overline{\theta}^i(t_{n(k)})$, let us call them $\overline{\theta}_\infty^i$ and $\overline{\phi}_\infty^i$, are equilibrium points of the ODE's in Lemma 4.8. These limits determine the long-term behavior of 3DPG.

Finally, the first part of Theorem 4.1 now follows (for the particular case of experience replay with size 1 and global information access without communication) using Assumption 4.2.3 and the

dominated convergence theorem to swap the order of differentiation and integration in $\tilde{\nabla} l^i$ and $\tilde{\nabla} g^i$. It is left to show Theorem 4.1 for 3DPG with experience replays and only local information access and to show that the limiting distributions $\overline{\mu}^i_\infty(s, \mathcal{A})$ are stationary with respect to the state Markov process. Both are the subject of the next section.

Extension to experience replays

Experience replay buffers play an important role in stabilizing RL algorithms in practice (Mnih, Kavukcuoglu, Silver, et al. 2015). The fundamental idea is to learn from past experiences to reduce an RL algorithm's bias towards an agent's interactions with its environment. For 3DPG, this means that at time n, an agent does not use the transition t^i_n to calculate the loss gradients, but it uses a random minibatch of past transitions t^i_m from old time steps $m \leq n$. Consequently, the training algorithm is not overly biased towards agents' interaction with the environment, reducing learning variance and thereby improving stability.

In the previous section, we analyzed 3DPG for centralized training where the global transitions t^i_n are locally available for every agent i. Additionally, we only used an experience replay of size one. To accommodate the use of experience replays in Section 4.4.1, the probability measure $\mu(t)$ needs to be redefined. In (4.9), each agent i samples $M < N$ global tuples independently from its (local) random experience replay $\mathcal{R}^i_n$ at every iteration n. The sampling processes of the agents will, in general, be different. Further, we will experience $\mathcal{R}^i_n \neq \mathcal{R}^j_n$, since the agents communicate the global tuples in a potentially delaying manner.

We now define a new measure process $\mu^i(t)$ for each agent. For $t \in [t_n, t_{n+1})$, define

$$\mu^i(t) \coloneqq \frac{1}{M} \sum_{j=1}^{M} \delta_{(s_{m(n,j,i)}, a_{m(n,j,i)})}. \tag{4.27}$$

Hence, $\mu^i(t)$ is the probability measure on $\mathcal{S} \times \mathcal{A}$ that places a mass of $1/M$ on each pair $(s_{m(n,j,i)}, a_{m(n,j,i)})$ for $1 \leq j \leq M$, where each $m(n, j, i)$ denotes one of the time indices sampled by agent i at time n from its memory $\mathcal{R}^i_n$. Notice that in the presence of communication and AoI, each experience replay is a random sequence of sets. If we use the redefined measures in (4.21) we get for every $t = t_n$ that

$$\begin{aligned} \tilde{\nabla} l^i(\theta^i(t), \phi(t), \mu^i(t)) &= \frac{1}{M} \sum_{j=1}^{M} \nabla_{\theta^i} l^i(\theta^i_n, \phi_n, s_{m(n,j,i)}, a_{m(n,j,i)}), \\ \tilde{\nabla} g^i(\theta^i(t), \phi(t), \mu^i(t)) &= \frac{1}{M} \sum_{j=1}^{M} \nabla_{\phi^i} g^i(\theta^i_n, \phi_n, s_{m(n,j,i)}). \end{aligned} \tag{4.28}$$

The analysis presented in Section 4.4.1 is also true for the new measure processes, where now every agent has its own measure process. This shows the first part of Theorem 4.1. It is left

to characterize the properties of the limiting measure processes μ^i_∞, which are the limits of the convergent subsequence extracted from $\mu^i_n(t) \coloneqq \mu^i(t_n + t)$.

Lemma 4.9. *For all $t \in [0, \infty)$ and for all agents i*

$$\mu^i_\infty(t, dy \times \mathcal{A}) = \int_{\mathcal{S}} p(dy \mid x, \phi_\infty(t))\mu^i_\infty(t, dx \times \mathcal{A}), \tag{4.29}$$

i.e., the limiting marginals constitute stationary distributions over the state Markov process.

To prove this lemma, we use Assumption 4.2.1(b), which describes the required trade-off between DataAoI growth of experience in the local replace memories $\mathcal{R}^i$ and the chosen stepsize sequences. Lemma 4.9 shows that the accumulated experiences of the agents constitute stationary distributions of the state-Markov process, provided that the DataAoI is not too large relative to the used stepsize. This is the announced answer to (Q3). We will now discuss further insides from the analysis for multi-agent actor-critic learning from old policies.

4.4.3 Cooperative training of MAS based on old actions vs. old policies

This section discusses the difference between 3DPG and MADDPG *for the centralized training scenario* with global information access.

The MADDPG policy gradient iteration

An MADDPG agent i updates its policy using the gradient

$$\frac{1}{M}\sum_m \nabla_{\phi^i} g^i_{\text{MADDPG}}(\theta^i_n, \phi^i_n, s_m, a^{j\neq i}_m) \tag{4.30}$$

for $1 \leq m \leq M$ sampled transitions, with $\nabla_{\phi^i} g^i_{\text{MADDPG}}$ as defined in (4.6). Please again observe the difference compared to the second iteration in the 3DPG algorithm (4.9). In MADDPG, old actions a^j_m for all $j \neq i$ are used from the samples of the experience replay. In contrast, the true current policy $a_j = \pi_{\phi^j_n}(s^j_m)$ is used in the 3DPG policy gradient iteration (assuming the centralized setting). MADDPG still uses the policies of other agents in the critic iteration. Hence, the availability of the policies from other agents is required in any way.

Let us try to understand the subtle difference between 3DPG and MADDPG intuitively. From the perspective of some agent i, the MADDPG policy gradient iteration appends the product action space of other agents $\prod_{j\neq i}\mathcal{A}_j$ to the global state space $\mathcal{S}$. Thus, averaging over the behavior of other agents occurs during training. For illustration, suppose agent i samples transitions $\{t_m\}_{m=1}^M$, and lets suppose all local states are equal, i.e. $s^i_m = s^i$. Then (4.30) gives $\nabla_{\phi^i}\pi_i(s^i; \phi^i_n)$ times

$$\left(\frac{1}{M}\sum_m \nabla_{a^i} Q^i(s^i, a^1_m, \ldots, \pi_i(s^i; \phi^i), \ldots, a^D_m)\right) \tag{4.31}$$

as the sample policy gradient. The expression averages over sampled actions of the other agents. These actions will sometimes include random actions due to exploration.[1] This indicates that agents would learn policies that also act well for random behavior of other agents. This seems to be undesirable for cooperative learning. We will now make the above heuristic precise using Theorem 4.1 and its variant for MADDPG.

The MADDPG limit vs. the 3DPG limit

First, we discuss the analog of Theorem 4.1 for MADDPG Specifically, consider (4.9) with $\nabla_{\phi^i} g^i_{\text{MADDPG}}$ as defined in (4.6) instead of $\nabla_{\phi^i} g^i_{\text{3DPG}}$ as defined in (4.7).

To analyze MADDPG with experience replays, we use the same measure processes (4.27) as in the Section 4.4.2. However, we need to redefine the average policy gradient in (4.21) to

$$\begin{aligned}\tilde{\nabla} g^i_{\text{MADDPG}}(\theta^i, \phi^i, \nu) \coloneqq \int \nabla_{\phi^i} g^i_{\text{MADDPG}}(\theta^i, \phi^i_n, s, a^{j\neq i}) \\ \nu(ds, da^1, \ldots, \mathcal{A}^i, \ldots, da^D).\end{aligned} \tag{4.32}$$

The analysis from Sections 4.4.1 and 4.4.2 can now be emulated for this gradient with the measure processes $\mu^i(t)$. Due to the new average policy gradient the conclusion of the convergence theorem are now fundamentally different. MADDPG converges to limits θ^i_{MA} and ϕ^i_{MA}, such that

$$\tilde{\nabla} g^i_{\text{MA}}(\theta^i_{\text{MA}}, \phi^i_{\text{MA}}, \mu^i_{\text{MA}}) = 0, \tag{4.33}$$

Here, μ^i_{MA} are the local limiting distribution of the sampled experience at agent i under MADDPG. Recall that the 3DPG limit satisfies:

$$\tilde{\nabla} g^i_{\text{3DPG}}(\overline{\theta}^i_\infty, \overline{\phi}_\infty, \overline{\mu}^i_\infty) = 0 \tag{4.34}$$

In (4.33), the behavior of the other agents is solely present in the limiting measures μ^i_{MA}. When exploration is stopped after some time, then the asymptotic properties of MADDPG and 3DPG are the same. However, when the exploration probability is not decayed to zero asymptotically, the limiting measures μ^i_{MA} are also shaped by random actions. This formalizes the heuristic from the previous subsection: the presence of exploration can deteriorate the policies found by MADDPG agents. The agents would adapt their policies to random actions that are not representative of the other agents. This is clearly an undesirable property. 3DPG fares better in this regard as it does not have this negative property! *3DPG allows a high exploration probability asymptotically without negatively affecting the found policies.* This claim is underpinned by a numerical experiment in Chapter 9.

[1] In most DeepRL algorithms, the probability of selecting a random action is decayed to a small value over time. However, it is usually kept positive to also allow some asymptotic exploration.

Remark 4.4.1. *In practice, the aforementioned negative property of MADDPG due to random actions is exacerbated because training is stopped after a finite time, possibly prematurely. During training, the other agents may have initially behaved in a certain way, which was then well represented in the replay memory of an agent. During later stages of learning, the other agents may "quickly" converge to a different policy. Since the agent uses outdated samples to calculate local gradients, the policy evolution is significantly biased towards the old behavior of the other agents. This is particularly undesirable in cooperative problems. Again, 3DPG circumvents such scenarios by using the latest available agent policies. MADDPG can overcome these issues by stopping exploration after some time and by using decaying learning rates.*

The moving target problem in MARL

Multi-agent RL algorithms are affected by non-stationarity due to the change of other agents' behavior from one agent's perspective (the so-called moving target problem) Canese et al. 2021. Here, 3DPG is no exception. The authors of (Lowe et al. 2017) discuss that using the local policy gradients $\nabla_{\phi^i} g^i_{\text{MADDPG}}$ based on old actions of other agents removes the non-stationarity from the perspective of each agent. We believe that this is not wholly true. MADDPG smooths out the non-stationarity, as the behavior of other agents is observed via the sampled actions from the experience replays. But this does not remove the non-stationarity. Intuitively, it takes longer for the behavior of an agent to manifest in the replay memory (in the form of samples) compared to directly using the behavior of an agent using its policy as in 3DPG.

Since 3DPG uses the policies of other agents, we expect that 3DPG will have more variance due to the changing behavior of other agents. However, since the 3DPG agents use more accurate information from the other agents' policies and are not affected by exploration as described in the previous section, we expect a faster convergence rate than MADDPG. Both predictions are supported by the experiment presented in Chapter 9. Finally, notice that the moving target problem has no impact on Corollary 4.2. The 3DPG agents converge asymptotically to a local Nash equilibrium with high probability.

4.5 Discussion and related work

We presented and analyzed 3DPG, a multi-agent reinforcement learning algorithm for decentralized, online learning in networked systems. We showed that the analysis can be modified to understand the popular MADDPG algorithm (Lowe et al. 2017). The analysis (and numerical examples in Chapter 9) show that 3DPG should be preferred when a multi-agent decision-making problem requires coordinated decisions or a high degree of exploration. More precisely, policy gradient training steps in multi-agent learning should preferably be based on past policies of other

agents since policy gradients evaluated with respect to random actions of other agents do not represent actual agent behavior and can thus negatively impact the policies found. For 3DPG, we showed that for 3DPG, the effect of unrepresentative behavior from old policies and potentially random actions does not affect the limiting policies found by 3DPG. Instead, for MADDPG, we showed that the presence of randomly explored actions can have a negative impact on the training result of MADDPG.

Beyond convergence characteristics, we presented the first set of data availability assumptions Assumption 4.2.1 for distributed online actor-critic learning in network systems. The assumptions describe how old locally available global data tuples can be so that 3DPG converges. More specifically, combined with the results to be presented in Chapter 5, arbitrary moment bounds for the AoI and DataAoI random variables will be sufficient under rapidly decaying stepsizes to guarantee convergence of 3DPG and convergence of the resulting state-Markov process. In addition, the numerical experiments in Chapter 9 will show that 3DPG is highly robust to using old information, making it attractive for distributed online multi-agent learning. An interesting direction for further investigation is to study the trade-off between using network resources for policy communication vs. using network resources for data communication.

Discussion of stability assumptions

Ideally, when one is dealing with a specific algorithm, it should be guaranteed or proven – rather than assumed up-front – that the algorithm iterations are stable. Assuming stability is, nonetheless, a typical first step toward understanding the convergence behavior of optimization algorithms. Especially in deep RL, the stability of algorithms like Deep Q-Learning or DDPG is not well understood. Most notably, there is a significant gap between the assumptions made in theory and assumptions verifiable in practice. Let us review results on the convergence of MAAC learning, since there are not that many. In (K. Zhang et al. 2018), the authors use linear function approximation and assume that the MA learning problem can be described by finite state ergodic Markov process. They further assume the existence of a projection operator with knowledge of a compact set that includes a local minima of the objective. (Kumar, Koppel, and Ribeiro 2019) provides a very interesting rate of convergence results for AC methods. However, they assume that samples (s_n, a_n, r_n, s_{n+1}) are drawn from a known stationary distribution of the state Markov process. Instead, we show that the AC iterates converge such that the agents' experience give rise to stationary distributions of the state Markov process. In addition, knowledge of the bias of the policy gradient and the bias of the critic estimates is required in (Kumar, Koppel, and Ribeiro 2019), while the critic should again be a linear combination of features. That work also assumes that the policy gradient is Lipschitz continuous, which would require Assumption 4.2.3(a) since most DNNs are only locally Lipschitz. Finally, recent work

by Y. Zhang et al. (2023) focuses on graph-based MARL settings with finite state and action spaces.

The assumptions made in the above works will be very hard to verify for most data-driven applications in practice. Even worse, we fear that guaranteeing stability for practical data-driven RL problems may always require assumptions that are not easily verifiable in practice. However, a practitioner may not even be highly interested in stability. Usually, practitioners will design their DNN parameterizations and their hyperparameter configurations using their experience, such that they roughly observe stable behavior. Afterward, practitioners want to know what limit they can expect from their algorithm. This is where this work comes into play.

In contrast to the assumptions made in the literature, the proposed assumptions, except Assumption 4.2.3(a), are very weak, easily verifiable in practice, and represent well how users apply DQN, DDPG, and its variants in practice. For this setting, this work answers where one can expect the 3DPG iterations (4.9) to converge asymptotically. Specifically, the analysis comprehensively characterizes the found limit using limiting distributions of the state-action process. These limiting distributions are shown to be stationary distributions of the state Markov process and are shaped by the experience of the agents.

Other related work

One algorithm that at first appears highly related (but only by its name) is the Distributed Distributional Deterministic Policy Gradients (D4PG) (Barth-Maron et al. 2018). This algorithm, however, is an algorithm for single-agent learning, and it uses parallel DDPG iterations with distributional critics.

4.6 Proofs of Chapter 4

Proof of Lemma 4.4. By Assumption 4.2.5 the neural network activations are twice-continuous differentiability (C^2), hence $\pi^i(s;\phi^i)$ and $Q^i(s,a;\theta^i)$ are C^2 in their input coordinates. Additionally, it follows from (Ramaswamy and Hullermeier 2021, Lemma 9) that $\pi^i(s^i;\phi^i)$ and $Q^i(s,a;\theta^i)$ are C^2 in their parameter coordinates ϕ^i and θ^i, respectively, for every fixed $s \in \mathcal{S}$ and $a \in \mathcal{A}$. Note that composition, product and sums of C^2 functions are C^2. Moreover C^2 functions have local Lipschitz gradients. This is because the gradient is C^1, and C^1 functions are locally Lipschitz (Conway 2019). This immediately shows that $\nabla_{\theta^i} l^i(\theta_n^i, \phi_n, s_n, a_n, s_{n+1})$ and $g(\theta_n^i, \phi_n, s_n)$ have the required properties.

For $\nabla_{\theta^i} \hat{l}^i(\theta_n^i, \phi_n, s_n, a_n)$, fix parameter vectors ϕ and θ^i as well as $s \in \mathcal{S}$ and $a \in \mathcal{A}$. Since, $\pi(s;\phi)$ and $Q^i(s,a;\theta^i)$ are C^2 in every coordinate, there is some $R > 0$ and continuous functions

$L_{Q^i}(y,\theta^i,\phi)$ and $L_\pi(y,\phi)$, such that $\forall\ \phi_1,\phi_2 \in \overline{B}_R(\phi)$, we have

$$\begin{aligned}
&\Big|\int Q^i(y,\pi(y;\phi_1);\theta^i)p(dy\mid s,a) - \int Q^i(y,\pi(y;\phi_2);\theta^i)p(dy\mid s,a)\Big| \\
&\leq \int L_{Q^i}(y,\theta^i)\|\pi(y;\phi_1)-\pi(y;\phi_2)\|_2 p(dy\mid s,a) \\
&\leq \|\phi_1-\phi_2\|_2 \int L_{Q^i}(y,\theta^i,\phi)L_\pi(y,\phi)p(dy\mid s,a) \\
&\leq L(\phi)\|\phi_1-\phi_2\|_2
\end{aligned} \tag{4.35}$$

for some $L(\phi)>0$. The last inequality follows from the stability of the critic iteration Assumption 4.2.3(a) and the compactness of the state space Assumption 4.2.3(b). Hence, $\nabla_{\theta^i}\hat{l}^i(\theta^i,\phi,s,a)$ is locally Lipschitz as a product and sum of locally Lipschitz functions. It is left to show that $\nabla_{\theta^i}\hat{l}^i(\theta^i,\phi,s,a)$ is continuous in the s and a coordinate. This directly follows from the convergence in distribution by continuity of $p(dy\mid s,a)$ and $r^i(s,a)$, Assumption 4.2.4 and Assumption 4.2.6 respectively, and since $Q^i(s,\pi(s;\phi);\theta^i)$ is a bounded continuous function by Assumption 4.2.3. □

Proof of Lemma 4.5. From Lemma 4.4, we have that $\nabla\hat{l}^i$ is locally Lipschitz. It follows from Assumption 4.2.3(a) that $\nabla\hat{l}^i$ is Lipschitz continuous with constant L when restricted to a sample path-dependent compact set. Using the triangular inequality, the established Lipschitz continuity of $\nabla\hat{l}^i$ and Assumption 4.2.3(a), it follows as in Chapter 2 that

$$\begin{aligned}
\|e_n^{\theta^i}\| &\leq L\sum_{j\neq i}\sum_{m=n-\tau_{ij}(n)}^{n-1}\|\phi_{m+1}^j-\phi_m^j\| \\
&\leq C\sum_{j\neq i}\sum_{m=n-\tau_{ij}(n)}^{n-1}\beta(m),
\end{aligned} \tag{4.36}$$

for a sample path-dependent constant $C>0$, which thus implies that $\lim_{n\to\infty}\|e_n^{\theta^i}\|=0$ by Assumption 4.2.1. The proof for $e_n^{\phi^i}$ follows analogously. □

Proof of Lemma 4.6. The proof is similar to the corresponding proof in (Ramaswamy and Hullermeier 2021) for deep q-learning, with some small changes due to the actor policies. We have that

$$\begin{aligned}
\psi_n^i = \gamma\Big(&Q^i(s_{n+1},\pi(s_{n+1};\phi_n);\theta_n^i) - \\
&\int Q^i(s,\pi(s;\phi_n);\theta_n^i)p(ds\mid s_n,a_n,\phi_n)\Big)\nabla_{\theta^i}Q^i(s_n,a_n;\theta_n^i).
\end{aligned} \tag{4.37}$$

Define the filtration $\mathcal{F}_{n-1} := \sigma(s_m,a_m,\theta_m,\phi_m\mid m\leq n)$ for $n\geq 1$. It then follows that $\{\Psi_n\}$ is a zero-mean martingale. It follows from Assumption 4.2.3 and the C^2 condition in Assumption 4.2.5 that $\sup_{n\geq 0}\|\psi_n^i\|\leq K<\infty$ for a sample path dependent constant K. It then follows from the martingale convergence theorem (see A.2) that Ψ_n^i converges, since $\sum_{m=0}^n\alpha^2(m)\|\psi_m^i\|^2<\infty$ almost surely by Assumption 4.2.1(a). □

Proof of Lemma 4.7. Fix $T > 0$. We define $[t]$ for $t \geq 0$ as $[t] \coloneqq t_{\sup\{n|t_n \leq t\}}$. Fix $t \in [0, T]$, then $[t_n + t] = t_{n+k}$ for some $k \geq 0$. Recall, that $\overline{\phi}^i(t)$ is defined by linear interpolation with respect to $\alpha(n)$, see (4.19). Hence, $\overline{\phi}^i(t_n + t) - \overline{\phi}^i(t_{n+k})$ is equal to

$$\frac{t_n + t - t_{n+k}}{\alpha(n+k)} \left(\overline{\phi}^i(t_{n+k+1}) - \overline{\phi}^i(t_{n+k}) \right). \tag{4.38}$$

The stability of the algorithm and the compactness of the state-action space, *i.e.* Assumption 4.2.3, show that $\nabla_{\phi^i} g^i$ is bounded and hence $\|\overline{\phi}^i(t_{n+k+1}) - \overline{\phi}^i(t_{n+k})\| \in \mathcal{O}(\beta(n+k))$. It follows that

$$\sup_{t \in [0,T]} \|\overline{\phi}^i(t_n + t) - \overline{\phi}^i([t_n + t])\| \in \mathcal{O}(\alpha(n)), \tag{4.39}$$

since $\alpha(n)$ is monotonic and $\frac{\beta(n)}{\alpha(n)} \to 1$. Similarly, we can show that

$$\sup_{t \in [0,T]} \|\phi_n^i(t) - \phi_n^i([t_n + t] - t_n)\| \in \mathcal{O}(\alpha(n)) \tag{4.40}$$

To show (4.25), we now need to show that

$$\sup_{t \in [0,T]} \|\overline{\phi}^i([t_n + t]) - \phi_n^i([t_n + t] - t_n)\| \to 0. \tag{4.41}$$

From (4.22) it follows that

$$\begin{aligned} &\|\overline{\phi}^i(t_{n+k}) - \phi_n^i(t_{n+k} - t_n)\| \leq \|\overline{\phi}^i(t_{n+k}) - \overline{\phi}^i(t_n) \\ &- \int_0^{t_{n+k}-t_n} \tilde{\nabla} g^i(\theta_n^i(x), \phi_n(x), \mu_n(x)) dx\|. \end{aligned} \tag{4.42}$$

Using a telescoping series, $\overline{\phi}^i(t_{n+k}) - \phi_n^i(t_{n+k} - t_n)$ equals

$$\begin{aligned} &\sum_{m=n}^{n+k-1} \beta(m) \nabla_{\phi^i} g^i(\theta_m^i, \phi_m, s_m) \\ &= \sum_{m=n}^{n+k-1} \int_{t_m}^{t_{m+1}} \frac{\beta(m)}{\alpha(m)} \tilde{\nabla} g^i(\overline{\theta}^i([x]), \overline{\phi}([x]), \mu_n(x - t_n)) dx \end{aligned} \tag{4.43}$$

The last step follows from $\alpha(m) = t_{m+1} - t_m$ and using that $\phi_m^i = \overline{\phi}^i(t_m) = \overline{\phi}^i([t])$ for all $t \in [t_m, t_{m+1})$. Now rewrite the second term in (4.42):

$$\begin{aligned} &\int_0^{t_{n+k}-t_n} \tilde{\nabla} g^i(\theta_n^i(x), \phi_n(x), \mu_n(x)) dx = \\ &\sum_{m=n}^{n+k-1} \int_{t_m}^{t_{m+1}} \tilde{\nabla} g^i(\theta_n^i(x - t_n), \phi_n(x - t_n), \mu_n(x - t_n)) dx \end{aligned} \tag{4.44}$$

We now evaluate the difference of the terms under the integrals in (4.43) and (4.44).

$$\begin{aligned}
&\|\frac{\beta(m)}{\alpha(m)}\tilde{\nabla}g^i(\overline{\theta}^i([x]),\overline{\phi}([x]),\mu_n(x-t_n)) - \tilde{\nabla}g^i(\theta_n^i(x-t_n),\phi_n(x-t_n),\mu_n(x-t_n))\| \\
&\leq C|\frac{\beta(m)}{\alpha(m)}-1|+\|\tilde{\nabla}g^i(\overline{\theta}^i([x]),\overline{\phi}([x]),\mu_n(x-t_n)) - \tilde{\nabla}g^i(\theta_n^i(x-t_n),\phi_n(x-t_n),\mu_n(x-t_n))\| \\
&\leq C|\frac{\beta(m)}{\alpha(m)}-1|+L\Big(\|\overline{\theta}^i([x]) - \overline{\theta}_n^i([x]-t_n)\| + \|\overline{\phi}([x]) - \phi_n([x]-t_n)\| \\
&\qquad + \|\theta_n^i(x-t_n) - \theta_n^i([x]-t_n)\| + \|\phi_n(x-t_n) - \phi_n([x]-t_n)\|\Big)
\end{aligned} \tag{4.45}$$

for some sample path dependent constant $C < \infty$ using the stability from Assumption 4.2.3. The last step adds zeros and uses the Lippschitz continuity of $\tilde{\nabla}g^i$. The combination of (4.42), (4.43), (4.44) and (4.45) thus gives:

$$\begin{aligned}
&\|\overline{\phi}^i(t_{n+k}) - \phi_n^i(t_{n+k}-t_n)\| \\
&\leq \sum_{m=n}^{n+k-1}\alpha(m)\mathcal{O}\left(\Big|\frac{\beta(m)}{\alpha(m)}-1\Big|\right) + L\sum_{m=n}^{n+k-1}\mathcal{O}\left(a(m)^2\right) \\
&+ L\sum_{m=n}^{n+k-1}a(m)\Big(\|\overline{\theta}^i(t_m) - \overline{\theta}_n^i(t_m-t_n)\| + \sum^i\|\overline{\phi}^i(t_m) - \overline{\phi}_n^i(t_m-t_n)\|\Big)
\end{aligned} \tag{4.46}$$

The first term in the above expression converges to zero as $n \to \infty$, since $\sum_{m=n}^{n+k-1}\alpha(m) \leq T$ by construction and since $\sum_{n\geq 0}(\alpha(n)-\beta(n)) < \infty$ from Assumption 4.2.1(b). The second term converges to zero since $\alpha(n)$ is square summable Assumption 4.2.1(a).

Inequality (4.46) can now be derived analogously for $\|\overline{\theta}^i(t_{n+k}) - \theta_n^i(t_{n+k}-t_n)\|$. We can now sum up all L.H.S. and R.H.S. for all i in (4.46), and for all θ^i:

$$\begin{aligned}
x_n &\coloneqq \sum_{i=1}^{D}\|\overline{\phi}^i(t_{n+k}) - \phi_n^i(t_{n+k}-t_n)\| \\
&+ \sum_{i=1}^{D}\|\overline{\theta}^i(t_{n+k}) - \theta_n^i(t_{n+k}-t_n)\| \\
&\leq o(1) + 2L\sum_{m=n}^{n+k-1}a(m)\sum_i\|\overline{\theta}^i(t_m) - \overline{\theta}_n^i(t_m-t_n)\| \\
&\qquad + 2LD\sum_{m=n}^{n+k-1}a(m)\sum_i\|\overline{\phi}^i(t_m) - \overline{\phi}_n^i(t_m-t_n)\|,
\end{aligned} \tag{4.47}$$

where D is the number of agents. We now apply the discrete version of Gronwall inequality (V. Borkar 2022) to x_n. It follows that $x_n \leq o(1)e^{2LD\sum_{m=n}^{n+k-1}a(m)}$. By construction $\sum_{m=n}^{n+k-1}a(m) \leq T$ for all $n \geq 0$, thus $x_n \to 0$, which proves the lemma.

□

Proof of Lemma 4.8. Consider the sequence θ_n^i. The proof for the other parameter sequences is identical. Fix $T > 0$. We need to show that

$$\sup_{t\in[0,T]} \|\theta_n^i(t) - \theta_\infty^i(0) - \int_0^t \tilde{\nabla} l^i(\theta_n^i(x), \phi_\infty(x), \mu_\infty(x))dx\| \tag{4.48}$$

converges to zero. Using (4.22), the norm in (4.48) is upper-bounded by

$$\|\theta_n^i(0) - \theta_\infty^i(0)\| + \|\int_0^t \tilde{\nabla} l^i(\theta_n^i(x), \phi_n(x), \mu_n(x)) - \tilde{\nabla} l^i(\theta_\infty^i(x), \phi_\infty(x), \mu_\infty(x))dx\|. \tag{4.49}$$

We can now expand the second term by successively adding zeros for each policy of each agent $j \neq i$. We can then use Lemma 4.4 to bound the resulting expansion from above by a term in

$$\mathcal{O}\left(\int_0^t \|\theta_n^i(x) - \theta_\infty^i(x)\| + \sum_{j\neq i} \|\phi_n^j(x) - \phi_\infty^j(x)\|dx\right), \tag{4.50}$$

Additionally, we are left with one term of the form

$$\|\int_0^t \tilde{\nabla} l^i(\theta_\infty^i(x), \phi_\infty(x), \mu_n(x)) - \tilde{\nabla} l^i(\theta_\infty^i(x), \phi_\infty(x), \mu_\infty(x))dx\|. \tag{4.51}$$

Due to the compact convergence of every parameter sequences (Arzela-Ascoli theorem) every parameter sequence will converge uniformly over $[0, T]$. This shows that (4.50) converges to zero. Finally, (4.51) converges to zero as $\mu_n \to \mu_\infty$ in distribution and since $\theta_\infty^i(t)$, $\phi_\infty(t)$ are bounded almost surely by Assumption 4.2.3(a) and Lemma 4.7. □

Proof of Lemma 4.9. Without loss of generality, assume a batch-size $M = 1$. The cases $M > 1$ will only require additional bookkeeping. Recall that the samples used in the 3DPG iterations (4.9) are potentially old and from random time-steps, such that Assumption 4.2.1(b) holds.[2] Fix some agent i. In the following, we will drop the agent index i. Since $M = 1$, the agent uses a global transition t_{k_n} with random time index k_n for its 3DPG training step at time n.

Pick $f \in C_b(\mathcal{S})$, the convergence determining class for distributions on $\mathcal{S}$. We analyze

$$\begin{aligned}&\int_{t_n}^{t_{n+1}} \left[f(s) - \int_{\mathcal{S}} f(y)p(dy \mid s, \phi_n)\right] \mu(z, ds, \mathcal{A})dz \\ &= \alpha(n)\left[f(s_{k_n}) - \int_{\mathcal{S}} f(y)p(dy \mid s_{k_n}, \phi_n)\right]\end{aligned} \tag{4.52}$$

as $n \to \infty$. The error terms (4.52) consider the deviation between states sampled from $\mu(t)$ and the associated expected transition under the *policy that uses the sample during training.* This is the perspective of an experimenter that observes the Markov game and the 3DPG algorithm during runtime.

[2]Notably, Assumption 4.2.1(b) in conjunction with the Borel-Cantelli lemma guarantee that infinitely many global transitions reach each agent.

Now accumulate the aforementioned deviations for all time steps where a sample t_n would be used during training and evaluate the deviation under the policy at time n. This information is of course not required for the algorithm and solely a quantity for the analysis. In summary, consider

$$\gamma(n)\left[f\left(s_{n+1}\right)-\int_{\mathcal{S}} f(y)p\left(dy \mid s_n, \phi_n\right)\right] \tag{4.53}$$

at time n, where

$$\gamma(n) \coloneqq \sum_{i\in\{i\geq 0 | k_i = n\}} \alpha(i). \tag{4.54}$$

for every $n \geq 0$ with $\sum_{i\in\emptyset} = 0$. Recall that $\alpha(n) \in \mathcal{O}(n^{-\frac{1}{q}})$ with $q \in [1, 2)$, then

$$\sum_{k=0}^{n}\gamma(k) \leq \mathcal{O}\left(\sum_{k=0}^{2n}\alpha(k)\right) \leq \mathcal{O}\left(\int_1^{2n} x^{-\frac{1}{q}}dx\right) \tag{4.55}$$

since at time n a sample from the replay memory can be at most n time steps old. It then follows that $\frac{1}{n}\sum_{k=0}^{n}\gamma(k) \in \mathcal{O}\left(n^{-\frac{1}{q}}\right)$, which in turn implies that $\gamma(k) \in \mathcal{O}\left(n^{-\frac{1}{q}}\right)$ and thus that $\gamma(m)$ is square summable.

To analyze (4.53), we now consider the sequence

$$\xi_n \coloneqq \sum_{m=0}^{n-1}\gamma(m)\left[f\left(s_{m+1}\right)-\int_{\mathcal{S}} f(y)p\left(dy \mid s_m, \phi_m\right)\right] \tag{4.56}$$

and the filtration

$$\mathcal{F}_{n-1} \coloneqq \sigma\langle s_m, a_m, \phi_m, \gamma(m-1) \mid m \leq n-1\rangle. \tag{4.57}$$

Then ξ_n is a Martingale with respect to $\mathcal{F}_{n-1}$[3]. Since f is bounded and $\gamma(m)$ is square summable, the quadratic variation process associated with the Martingale ξ_n is convergent. It then follows from the martingale convergence theorem (see A.2) that ξ_n converges almost surely.

Recall that we denote the DataAoI by $\Delta(n)$, which is the age of the oldest sample in the replay memory. In other words,

$$k_n \in [n - \Delta(n)], \text{ for all } n \geq 0. \tag{4.58}$$

Since ξ_n converges and $\sum_{k=n-\Delta(n)}^{n-1}\alpha(k) \to 0$ a.s. by Assumption 4.2.1(b), it follows that for every $t > 0$

$$\sum_{m=n-\Delta(n)}^{\delta(n,t)}\gamma(m)\left[f\left(s_{m+1}\right)-\int_{\mathcal{S}} f(y)p\left(dy \mid s_m, \phi_m\right)\right] \tag{4.59}$$

converges to zero a.s., where

$$\delta(n,t) \coloneqq \min\{m \geq n \mid t_m \geq t_n + t\}.$$

[3]Technically, this assumes that a transition does not impact when previous transitions become available for training, i.e., independence of the Markov game and the communication process.

Next, spread out and rearrange the aggregated samples in (4.59). Specifically, use (4.58) and separate the samples as the following three terms, whose sum converges to zero a.s.:

$$\begin{aligned}&\sum_{m=n}^{\delta(n,t)} \alpha(m) \left[f\left(s_{k_m+1}\right) - \int_{\mathcal{S}} f(y) p\left(dy \mid s_{k_m}, \phi_{k_m}\right) \right] \\ &+ \mathcal{O}\left(\sum_{m=n-\Delta(n)}^{n} \alpha(m) \right) + \mathcal{O}\left(\sum_{m=\delta(n,t)}^{\delta(n,t)+\Delta(\delta(n,t))} \alpha(m) \right)\end{aligned} \tag{4.60}$$

The rearrangement is mathematically valid since (4.59) is a finite sum for each n. By Assumption 4.2.1, it follows that the second and third terms converge to zero almost surely. Hence, we conclude that

$$\sum_{m=n}^{\delta(n,t)} \alpha(m) \left[f\left(s_{k_m+1}\right) - \int_{\mathcal{S}} f(y) p\left(dy \mid s_{k_m}, \phi_{k_m}\right) \right] \tag{4.61}$$

converges to zero a.s.

Equation (4.61) now also holds, if we replace ϕ_{k_m} by ϕ_m since the resulting error terms when taking the difference between (4.61) and the version with ϕ_m converges to zero. To see this, note that $\sum_{m=n}^{\delta(n,t)-1} \alpha(m) \in \mathcal{O}(t)$ by construction. Further, all individual error terms in the aforementioned difference converge to zero using weak convergence by continuity of $(s, \phi) \mapsto p(\cdot \mid s, \phi)$ and since $\|\phi_{k_m} - \phi_m\| \to 0$ a.s. Finally, $\alpha(n)$ is eventually decreasing, hence

$$\sum_{m=n}^{\delta(n,t)} \left[\alpha(m) - \alpha(m+1)\right] f(s_{k_m+1}) \to 0 \text{ a.s.} \tag{4.62}$$

In summary, we have thus shown that

$$\sum_{m=n}^{\delta(n,t)} \alpha(m) \left[f\left(s_{k_m}\right) - \int_{\mathcal{S}} f(y) p\left(dy \mid s_{k_m}, \phi_m\right) \right] \tag{4.63}$$

converges to zero a.s. With (4.52) and (4.63) it then follows that

$$\int_{t_n}^{t_n+t} \int_{\mathcal{S}} \Big[f\left(s\right) - h(z,s) \Big] \mu(z, ds, \mathcal{A}) dz \to 0 \text{ a.s.} \tag{4.64}$$

where $h(z,s) := \int_{\mathcal{S}} f(y) p\left(dy \mid s, \overline{\phi}(z)\right)$. The lemma now follows from (4.64). We refer to (Ramaswamy and Hullermeier 2021, Lemma 6) for details on this final step. □

Part II: Age of Information Processes

Chapter 5

Stochastic Information Delays

AoI sequences, such as $\tau_{ij}(n)$ in iteration (1.2), (2.1) or (4.9), are usually not equipped with a certain process structure. It is common to assume that AoI sequences are either bounded (Agarwal and Duchi 2011) or bounded by deterministic sequences (Zhou et al. 2022). Instead, we view $\tau_{ij}(n)$ as a discrete-time stochastic process that describes what information is used to update an iteration. In single-path communication scenarios between two systems, it is evident that AoI sequences satisfy the unit growth property $\tau_{ij}(n+1) \leq \tau_{ij}(n) + 1$; an iteration can at most be updated ones per unit time step. On the other hand, in asynchronous computing scenarios with a single component as in Example 1.0.1, the AoI increment $\tau_{ij}(n+1) - \tau_{ij}(n)$ is usually in the order of the number of computing nodes assigned to update parts of a parameter space. This chapter uses AoI sequences with bounded increments as one characterizing property to define *Age of Information Processes*, which answers (Q5).

5.1 AoI processes: A definition

We will now define simple AoI processes and AoI processes based on the observations that 1) information exchange between two systems has AoI with unit or bounded increments and 2) AoI in asynchronous computing can be represented by a process taking values of many such processes. For simple AoI processes, we allow the discrete-valued process to have bounded increments.

Definition 5.1.1. *A **simple Age of Information Process** is a discrete-time stochastic process $\{\tau(n)\}_{n\geq 0}$ on the non-negative integers associated with another stochastic process $\{X_n\}$:*

1) $x_{n-\tau(n)}$ is the sample used by another system/iteration/agent at time n,
2) $\sup_{n\geq 0} \tau(n+1) - \tau(n) < \infty$ a.s.

We will now state the definition of what we call an AoI process.

Definition 5.1.2. *An **Age of Information Process** is a discrete-time stochastic process $\tau(n)$ associated with potentially time-varying family $\mathcal{K}(n) \subset \mathbb{N}$ of simple AoI processes $\{\tau_k(n)\}$, such that*

$$\tau(n) := \sum_{k \in \mathcal{K}(n)} \mathbb{1}_{\{I(n)=k\}} \tau_k(n) \tag{5.1}$$

for some integer-valued random process $I(n)$.

In Definition 5.1.2, the integer-valued random process $I(n)$ selects which of the simple AoI processes is "active" at step n. Further, the set $\mathcal{K}(n)$ represents which of the simple AoI processes can potentially be active at step n. For asynchronous computing, this means that scheduling and rescheduling of workers can be represented by Definition 5.1.2.

5.2 Deterministic growth properties from AoI moment bounds

With AoI processes defined, we can now study the connection between AoI moment bounds and deterministic almost sure AoI growth bounds. Specifically, we provide tools to extract deterministic growth bounds from moment bounds. Such deterministic growth bounds will then be used to verify the AoI-stepsize condition such as (1.12) for stability and convergence of distributed SA algorithms. The first result is a deterministic growth bound for simple AoI processes.

Lemma 5.1. *Let $\tau(n)$ be a simple AoI process. Suppose there exists a random variable $\overline{\tau}$ with $\tau(n) \leq_{st} \overline{\tau}$ for all $n \geq 0$ with $\mathbb{E}[\overline{\tau}^p]$ for some $p \in (0, \infty)$, then for all $\varepsilon > 0$,*

$$\mathbb{P}\left(\tau(n) > \varepsilon n^{\frac{1}{\max\{1,p\}}} \text{ i.o.}\right) = 0. \tag{5.2}$$

Proof sketch. For $p > 1$, the lemma follows from the first Borel-Cantelli Lemma (see A.2) and stochastic dominance (V. Borkar 2022, Sec. 6). For $p \in (0, 1]$, first, use the Borel-Cantelli Lemma and the stochastic dominance property to conclude that $\tau(n)$ has a subsequence that does not exceed any fraction of n after some time. Then, use the bounded increment property to show that $\tau(n)$ does not exceed any fraction of n plus a term in the order of n^{1-p} after some time. A simple argument based on the limit superior then completes the proof. □

Using basic properties of the limit supremum and assuming that the number of possible active simple AoI processes at every point in time is finite, we conclude with the corresponding growth bound for AoI processes. This growth bound is the core answer to (Q6).

Lemma 5.2. *Let $\tau(n)$ be an AoI process with time-varying family $\mathcal{K}(n)$, such that for each $\tau_k(n)$ there exist some $p_k > 0$ with $\mathbb{P}\left(\tau_k(n) > \varepsilon n^{\frac{1}{\max\{1,p_k\}}} \text{ i.o.}\right) = 0$ for all $\varepsilon > 0$. If $\sup_{n \geq 0} |\mathcal{K}(n)| < \infty$ and $p := \inf_k p_k > 0$, then*

$$\mathbb{P}\left(\tau(n) > \varepsilon n^{\frac{1}{\max\{1,p\}}} \text{ i.o.}\right) = 0.$$

5.3 Convergence of accumulated stepsizes over AoI horizons

The next result formulates the trade-off between a moment bound for an AoI process and the decay of an algorithm stepsize sequence $a(n)$. A faster uniform tail decay of the AoI distributions allows slower decaying stepsize and, hence, potentially faster convergence. In particular, if the statement of Lemma 5.2 holds with $p > 1$, then stepsizes that decay slower than $\frac{1}{n+1}$ become available to satisfy Assumption 2.1.5 and Assumption 4.2.1, in Chapter 2 and Chapter 4, respectively. *Notably, the standard stepsize* $a(n) = \frac{c}{n+1}$, $n \geq 0$, $c > 0$, *turns out to be sufficient for stability and convergence of the distributed SA iterations discussed herein under an arbitrary AoI process moment bound and most notably for* $p \in (0, 1]$.

Lemma 5.3. *Consider an AoI process such that Lemma 5.2 holds for some* $p > 0$. *If the stepsize* $a(n) \in \mathcal{O}\left(n^{-\frac{1}{\max\{1,p\}}}\right)$, *then*

$$\sum_{k=n-\tau(n)}^{n-1} a(k) \to 0 \text{ a.s.} \tag{5.3}$$

Proof. Consider $p \in (0, 1]$ and without loss of generality pick $a(n) = \frac{1}{n+1}$. Since $a(n)$ is monotonically decreasing it follows that

$$\sum_{k=n-\tau(n)}^{n-1} a(k) \leq \frac{1}{n-\tau(n)} + \int_{n-\tau(n)}^{n-1} \frac{1}{t+1} dt = \frac{1}{n-\tau(n)+1} + \log\left(\frac{n}{n-\tau(n)+1}\right). \tag{5.4}$$

Then the lemma follows by continuity of the logarithm provided that

$$\frac{n-\tau(n)+1}{n} \to 1 \text{ a.s.} \iff \frac{\tau(n)}{n} \to 0 \text{ a.s.}. \tag{5.5}$$

To show this, we state a lemma that is often used to prove the strong law of large numbers.

Lemma 5.4 ((K. L. Chung 2001, Lemma 4.2.2.)). *Let* $X_1, X_2, \ldots$ *be real-valued random variables. Suppose for each* $\varepsilon > 0$, $\mathbb{P}(|X_n| \geq \varepsilon \text{ i.o.}) = 0$. *Then* X_n *converges to 0 almost surely.*

By assumption there is a random variable $\overline{\tau}$ with $\tau(n) \leq_{\text{st}} \overline{\tau}$ and $\mathbb{E}[\overline{\tau}^p] < \infty$ with $p \in (0, 1]$. Lemma 5.2 thus shows that $\mathbb{P}\left(\tau(n)n^{-1} > \varepsilon \text{ i.o.}\right) = 0$ for every $\varepsilon > 0$. Hence, the statement follows from Lemma 5.4. The case $p > 1$ follows the same line of argument. □

With Lemma 5.2, we have now established verifiable conditions for Theorem 2.1, Theorem 3.1 and Theorem 4.1, providing further details on the answer of (Q1) and (Q2). The verifiable conditions require the existence of dominating random variables with some arbitrary moment bound. The next chapter will show that the existence is guaranteed for AoI processes driven by strongly mixing event processes. Afterward, Chapter 7 will derive dominating random variables with prescribed moment bounds for asynchronous computing scenarios modeled as parallel point processes.

We will now close this chapter with the proof of Theorem 3.6 from Chapter 3. Recall that we established the almost sure rate of convergence of DASGD in Theorem 3.1 as

$$\mathcal{O}\left(\frac{1+\sum_{i,j}\sum_{k=0}^{n} a(k)\sum_{m=k-\tau_{ij}(k)}^{k-1} a(m)}{\sum_{k=0}^{n} a(k)}\right). \tag{5.6}$$

To make this rate of convergence estimate more precise, we need a rate of convergence estimate for $\sum_{k=n-\tau(n)}^{n-1} a(k)$. We will do this given that $\tau_{ij}(n)$ are stochastically dominated by random variable $\overline{\tau}$ with $\mathbb{E}\left[\overline{\tau}^p\right] < \infty$ for some $p > 1$.

Proof of Theorem 3.6. Let $a(n) \in \mathcal{O}\left(n^{-q}\right)$ for some $q \in (1/2, 1)$ and consider a dominating random variable with $\overline{\tau}$ with $\mathbb{E}\left[\overline{\tau}^p\right] < \infty$ for some $p > 1$. Then

$$\sum_{k=n-\tau_{ij}(n)}^{n-1} a(k) \in \mathcal{O}\left((n-1)^{1-q} - (n-\tau_{ij}(n))^{1-q}\right) \in \mathcal{O}\left(\tau_{ij}(n) n^{-q}\right) \in \mathcal{O}\left(n^{\frac{1}{p}-q}\right) \tag{5.7}$$

The second inclusion can be seen from a Taylor expansion at $n = \infty$. The third inclusion follows as $\mathbb{P}\left(\tau_{ij}(n) > n^{\frac{1}{p}} \text{ i.o.}\right) = 0$, which follows from $\mathbb{E}\left[\overline{\tau}^p\right] < \infty$ and the Borel-Cantelli lemma. The Theorem now follows from a minimization of (5.6) over q using (5.7). □

Remark 5.3.1 (Sufficient condition to apply Lemma 5.3)**.** *The existence of a dominating random variable with bounded p-th moment is guaranteed if $\tau_{ij}(n)$ are bounded in $L^{p'}$ for some $p' > p$, i.e., if $\sup_{n\geq 0} \mathbb{E}\left[\tau_{ij}^{p'}(n)\right] < \infty$. A proof of this implication can be found in (Leskelä and Vihola 2013, Proposition 3). We will directly establish the existence of dominating random variables for asynchronous computing and mobile wireless communication in Chapters 7 and 8, respectively.*

Remark 5.3.2 (Comment on stochastic dominance)**.** *It is always possible to construct a random variable that stochastically dominates a finite set of random variables, provided that stochastically dominating random variables are given for each random variable of the finite set. Hence, assuming a single dominating random variable for all AoI processes in Lemma 5.3 is without any loss in generality.*

Remark 5.3.3 (Importance of $p \in (0, 1]$ in Lemma 5.2 and Lemma 5.3)**.** *As mentioned in the introduction, it is new to the literature that an arbitrary moment bound for the AoI sequences is sufficient for the stability and convergence of SAs. This is important, most notably due to the newly explored territory with moment bounds for $p \in (0, 1]$. Such scenarios especially occur in asynchronous parallel computing due to job resource requirements that are heavy-tailed (Tirmazi et al. 2020; Samsi et al. 2021). In Chapter 7, we will characterize how the cases $p \in (0, 1]$ arise due to asynchronous computing.*

5.4 Proofs of Chapter 5

Proof of Lemma 5.1. Fix, $\varepsilon \in (0,1)$. Using the stochastic dominance property, it follows that

$$\sum_{n\geq 0} \mathbb{P}\left(\tau(n^{\frac{1}{p}}) > \varepsilon n^{\frac{1}{p}}\right) \leq \sum_{n\geq 0} \mathbb{P}\left(\overline{\tau} > \varepsilon n^{\frac{1}{p}}\right) = \sum_{n\geq 0} \mathbb{P}\left(\frac{1}{\varepsilon^p}\overline{\tau}^p > n\right) \leq 1 + \frac{1}{\varepsilon^p}\mathbb{E}\left[\overline{\tau}^p\right]. \tag{5.8}$$

The last inequality can be found in (K. L. Chung 2001, Theorem 3.2.1). Since, $\mathbb{E}\left[\tau^p\right] < \infty$ it follows from the first Borel-Cantelli lemma that $\tau(n^{\frac{1}{p}}) \leq \varepsilon n^{\frac{1}{p}}$ for all $n \geq N(\varepsilon)$ with a sample path dependent constant $N(\varepsilon) \geq 0$. In other words,

$$\tau(n) \leq \varepsilon n, \qquad \text{for } n = k^{\frac{1}{p}} \text{ with } k \geq N(\varepsilon). \tag{5.9}$$

Next, consider two subsequent integers n' and n'' that satisfy (5.9). Specifically, let $k \geq N(\varepsilon)$ and fix $n' = k^{\frac{1}{p}}$ and $n'' = (k+1)^{\frac{1}{p}}$. By the almost sure bounded increments, it follows that the AoI of the time steps between n' and n'' satisfy

$$\tau(n'+i) \leq \tau(n') + C_1 i \leq \varepsilon n' + C_1 i \tag{5.10}$$

for a sample path dependent constant $C_1 \geq 1$ and for every $i \in \{0,\dots,n''-n'\}$. Further, $i \leq C_2 k^{\frac{1}{p}-1}$ for a constant $C_2 > 0$ that only depends on p.

To see that C_2 does not depend on p, consider $k > 1$ and use Newton's generalized binomial theorem (Graham et al. 1989, p. 162). Then,

$$|(k+1)^{\frac{1}{p}} - k^{\frac{1}{p}}| = |\sum_{s=0}^{\infty} \binom{\frac{1}{p}}{s} k^{\frac{1}{p}-s} - k^{\frac{1}{p}}| = k^{\frac{1}{p}}|\left(\sum_{s=0}^{\infty} \binom{\frac{1}{p}}{s} k^{-s} - 1\right)| \tag{5.11}$$

$$= k^{\frac{1}{p}}|\left(\sum_{s=1}^{\infty} \binom{\frac{1}{p}}{s} k^{-s}\right)| \leq k^{\frac{1}{p}}\left(\sum_{s=1}^{\infty} |\binom{\frac{1}{p}}{s}| k^{-s}\right) \tag{5.12}$$

$$\leq \sup_{s\geq 1}|\binom{\frac{1}{p}}{s}| k^{\frac{1}{p}} \sum_{s=1}^{\infty} k^{-s} = \sup_{s\geq 1}|\binom{\frac{1}{p}}{s}| k^{\frac{1}{p}}(k-1)^{-1} \tag{5.13}$$

with

$$\binom{\frac{1}{p}}{s} = \frac{\frac{1}{p}\left(\frac{1}{p}-1\right)\left(\frac{1}{p}-2\right)\dots\left(\frac{1}{p}-s+1\right)}{s!}. \tag{5.14}$$

Notice that $\sup_{s\geq 1}|\binom{\frac{1}{p}}{s}|$ is bounded, since $|\binom{\frac{1}{p}}{s}| \leq 1$ for $s \geq \frac{1}{p}$. We could now define $C \coloneqq 3\sup_{s\geq 1}|\binom{\frac{1}{p}}{s}|$, since $(k-1)^{-1} < 3k^{-1}$ for all integers $k > 1$.

Next, with $k = (n')^p$, it follows that

$$\tau(n'+i) \leq \varepsilon n' + C_1 C_2 (n')^{1-p} \tag{5.15}$$

Since this holds for all pairs n', n'' with $i \in \{0,\dots,n''-n'\}$, it follows that $\tau(n) \leq \varepsilon n + C_1 C_2 n^{1-p}$ for all $n \geq N(\varepsilon)^{\frac{1}{p}}$. We have therefore shown that

$$\mathbb{P}\left(\tau(n) > \varepsilon n + C_1 C_2 n^{1-p} \text{ i.o.}\right) = 0. \tag{5.16}$$

for every $\varepsilon \in (0,1)$. Now fix $\varepsilon' \in (0,1)$ and let $\varepsilon = \frac{1}{2}\varepsilon'$. Then $C_1 C_2 n^{1-p} \leq \varepsilon n$ for all $n \geq N$ for some $N \in \mathbb{N}$. Hence, $\{\tau(n) > \varepsilon n + C_1 C_2 n^{1-p}\} \supset \{\tau(n) > 2\varepsilon n\}$ for $n \geq N$. By definition of the limit superior, it then follows from (5.16) that

$$\mathbb{P}\left(\tau(n) > \varepsilon' n \text{ i.o.}\right) \leq \mathbb{P}\left(\tau(n) > \varepsilon n + (1-\varepsilon) C n^{1-p} \text{ i.o.}\right) = 0. \tag{5.17}$$

□

Proof of Lemma 5.2. Fix $\varepsilon > 0$. From the definition of the limit supremum, we have that

$$\begin{aligned}
\mathbb{P}\left(\tau(n) > \varepsilon n^{\frac{1}{\max\{1,p\}}} \text{ i.o.}\right) &= \lim_{m\to\infty} \mathbb{P}\left(\bigcup_{n\geq m} \{\tau(n) > \varepsilon n^{\frac{1}{\max\{1,p\}}}\}\right) && (5.18)\\
&\leq \lim_{m\to\infty} \mathbb{P}\left(\bigcup_{n\geq m} \bigcup_{k\in\mathcal{K}(n)} \{\tau_k(n) > \varepsilon n^{\frac{1}{\max\{1,p\}}}\}\right) && (5.19)\\
&\leq \lim_{m\to\infty} \mathbb{P}\left(\bigcup_{k\in\mathcal{K}(n)} \bigcup_{n\geq m} \{\tau_k(n) > \varepsilon n^{\frac{1}{\max\{1,p_k\}}}\}\right) && (5.20)\\
&\leq \sum_{k\in\mathcal{K}(n)} \lim_{m\to\infty} \mathbb{P}\left(\bigcup_{n\geq m} \{\tau_k(n) > \varepsilon n^{\frac{1}{\max\{1,p_k\}}}\}\right) && (5.21)\\
&= \sum_{k\in\mathcal{K}(n)} \mathbb{P}\left(\tau_k(n) > \varepsilon n^{\frac{1}{\max\{1,p_k\}}} \text{ i.o.}\right) = 0. && (5.22)
\end{aligned}$$

The second inequality uses the definition of AoI processes; the third inequality uses that $p \leq p_k$ for all k and swaps the order of the unions; the fourth inequality uses the union bound for probabilities and that $\sup_n \mathcal{K}(n) < \infty$. □

Chapter 6

AoI arising from strongly mixing event processes

In the last chapter, we saw how the existence of AoI dominating random variables enables the stability and convergence results from Chapters 2 to 4. To verify the existence of AoI dominating random variables, this chapter studies the fundamental AoI processes (1.8), which we recall here for easy reference:

$$\tau(n+1) \coloneqq \begin{cases} 1, & A(n), \\ \tau(n)+1, & A(n)^c, \end{cases} \tag{6.1}$$

with $\tau(1) \coloneqq 1$. The fundamental AoI process measures the freshness of status updates from a source available at a monitor. An example sample path was illustrated in Figure 1.3. We will characterize properties for $\tau(n)$ based on assumptions on the associated event sequence $A(n)$. The results are based on (Redder, Ramaswamy, and Karl 2022b).

6.1 Main statements

We study the abstract AoI process (6.1) in the fundamental setting where the event sequence $A(n)$ merely possesses a dependency decay property. The process dependency decays over time, described by the α-mixing notion, which will be formally stated in the next section. Due to this abstract view, the developed results can be applied to AoI processes that arise in communication or computing or as a consequence of other phenomena that delay information transport. The results relate the so-called α-mixing rate of the event sequence $A(n)$ to properties of the AoI process $\tau(n)$. The α-mixing determines how the dependency of $A(n)$ and $A(m)$ decays as $|n-m| \to \infty$. Based on this rate, we calculate moment bounds for the AoI process $\tau(n)$ as a function of the α-mixing rate. Further, we show that $\tau(n)$ is itself α-mixing, which leads to a strong law of large numbers (SLLN) for $\tau(n)$ under sufficiently rapid α-mixing of the event sequence $A(n)$.

The first result is a set of sufficient conditions guaranteeing that the average p-th moment of the AoI process (6.1) is finite. We will use $\alpha(A,n)$ to denote the α-mixing coefficients of the indicator process $\mathbb{1}_{A(n)}$. Theorem 6.1 shows that sufficiently fast decay of $\alpha(A,n)$ guarantees that all AoI random variables $\tau(n)$ are stochastically dominated by a single random variable that has certain finite moments depending on the decay rate of $\alpha(A,n)$.

Theorem 6.1. *Suppose there are $\kappa \geq 0$ and $\varepsilon \in (0,1)$ such that $\mathbb{P}\left(\bigcup_{k=n}^{n+\kappa} A(k)\right) \geq \varepsilon$ for all $n \geq 0$. If $\sum_{m\geq 0} m^{p-1}\alpha(A,m) < \infty$ for some $p > 0$, then there is a random variable $\overline{\tau}$ with $\tau(n) \leq_{st} \overline{\tau}$ for all $n \geq 0$ and $\mathbb{E}\left[\overline{\tau}^p\right] < \infty$.*

The p-th moment of the dominating random variable $\overline{\tau}$ is calculated in the proof of Theorem 6.1 as a function of ε, κ and the mixing coefficients $\alpha(A,n)$. Theorem 6.1 is the core answer to (Q7) and describes how fast the dependency of events has to decay to guarantee AoI dominating random variables with prescribed moment bounds.

Notice that Theorem 6.1 does not require that $A(n)$ is a stationary process. Hence, $\frac{1}{N}\sum_{n=0}^{N-1}\tau^p(n)$ might not converge. However, by stochastic dominance, we have a bound for the limiting average moments of the AoI process:

Corollary 6.2. *If Theorem 6.1 holds for some $p > 0$, then*

$$\limsup_{N\to\infty} \mathbb{E}\left[\frac{1}{N}\sum_{n=0}^{N-1}\tau^p(n)\right] \leq \mathbb{E}\left[\overline{\tau}^p\right]. \tag{6.2}$$

Secondly, we study the mixing rate of the AoI process (6.1) itself. Here, the main result bounds the mixing rate of (6.1) by a combination of the mixing coefficient $\alpha(A,n)$ and the tail decay of the dominating random variable from Theorem 6.1.

Theorem 6.3. *Suppose the assumptions of Theorem 6.1 are satisfied, then the AoI process $\tau(n)$ is α-mixing with coefficients $\alpha(\tau,n)$, such that*

$$\alpha(\tau,n) \leq \min_{0\leq m\leq n}\left\{\alpha(A,n-m) + \mathbb{P}\left(\overline{\tau} > m\right)\right\} \tag{6.3}$$

with $\overline{\tau}$ from Theorem 6.1.

We also show that the mixing rate of $\alpha(\tau,n)$ is almost as fast as the mixing rate of $\alpha(A,n)$. Specifically, we show that

$$\sum_{n=0}^{\infty} n^{p-1}\alpha(A,n) < \infty \implies \sum_{n=0}^{\infty} n^{q-1}\alpha(\tau,n) \quad \forall q < p. \tag{6.4}$$

Using (6.3), we conclude Section 6.3.2 with a SLLN for the AoI process (6.1) under sufficiently rapid α-mixing of the event sequence $A(n)$. Specifically, under the conditions of Theorem 6.1 for some $p > 1$, we obtain the SLLN

$$\frac{1}{N}\sum_{n=0}^{N-1}\left(\tau(n) - \mathbb{E}\left[\tau(n)\right]\right) \xrightarrow{\text{a.s.}} 0. \tag{6.5}$$

6.2 Strong mixing and assumptions

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be the underlying probability space and let $\mathcal{A}$ and $\mathcal{B}$ be two sub-σ-algebras of $\mathcal{F}$. The following is the arguably most natural measure of dependency between $\mathcal{A}$ and $\mathcal{B}$

$$\alpha(\mathcal{A}, \mathcal{B}) \coloneqq \sup_{A \in \mathcal{A}, B \in \mathcal{B}} |\mathbb{P}(A \cap B) - \mathbb{P}(A)\,\mathbb{P}(B)|. \tag{6.6}$$

Consider a (not necessarily stationary) stochastic process $X = \{X(n)\}_{n \geq 0}$. For $0 \leq l \leq m \leq \infty$, define the sub-$\sigma$-algebra generated from $X(l)$ up to $X(m)$ by

$$\mathcal{F}_l^m \coloneqq \sigma\left(X(n) \mid l \leq n \leq m\right), \tag{6.7}$$

Informally, the σ-algebra generated by a stochastic process from a time interval describes the information that can be extracted from the associated process realizations (Durrett 2019). With these σ-algebras, we can now define *α-mixing*, which is a notion of asymptotic independence. We refer to Bradley (2005) for a survey about α-mixing and other mixing notions.

Definition 6.2.1. *The α-mixing coefficients of the process X are*

$$\alpha(X, n) \coloneqq \sup_{l \geq 0} \alpha(\mathcal{F}_0^l, \mathcal{F}_{l+n}^\infty), \quad n \geq 0. \tag{6.8}$$

The process X is called α-mixing *(or just strongly mixing) if $\alpha(X, n) \to 0$ as $n \to \infty$.*

We are now ready to formalize the assumptions for (6.1).

Recall that successful status updates from the source node are received at the monitor whenever an event $A(n)$ occurs. Hence, whenever an event $A(n)$ occurs, the monitor receives a fresh update during time slot n, thus $\tau(n+1) = 1$. We refer to $A(n)$ as the *event process* of the AoI process $\tau(n)$. We make the following assumptions for the event process $A(n)$:

Assumption 6.2.1. *There is some $\varepsilon \in [0, 1)$, such that*

$$\mathbb{P}(A(n)) \geq 1 - \varepsilon, \quad \forall n \geq 0.$$

Assumption 6.2.2. *The event process $A(n)$ is α-mixing (Definition 6.2.1) with coefficients $\alpha(A, n)$.*

Assumption 6.2.1 requires that at every time step, the monitor receives an update from the source with non-zero probability. A slightly weaker assumption that is also sufficient for the results is that there is some $\kappa \geq 0$ and some $\varepsilon \in [0, 1)$, such that $\mathbb{P}\left(\bigcup_{k=n}^{n+\kappa} A(k)\right) \geq 1 - \varepsilon, \quad \forall n \geq 0$. The weaker version thus requires that for every time interval of the form $[n, n+\kappa]$, the probability that the monitor receives an update from the source is greater than zero. This weaker assumption is, in fact, necessary for the existence of a random variable $\overline{\tau}$ such that $\tau(n) \leq_{\text{st}} \overline{\tau}$ for all $n \geq 0$; without it, there exists a subsequence $\{n_k\}$ such that $\mathbb{P}\left(\lim_{k \to \infty} \tau(n_k) = \infty\right) = 1$. For simplicity, we will present the proofs of Section 6.3.1 and Section 6.3.2 for $\kappa = 0$, though all proofs can be easily extended to $\kappa > 0$.

6.3 Moment bounds, mixing rates, and strong law

In this section, we analyze the moments of the AoI process (6.1). As the event process $A(n)$ may generally be non-stationary, we use stochastic dominance to construct uniformly dominating random variables for the AoI process $\tau(n)$. This immediately leads to bounds for the limiting average moments of the AoI process. Furthermore, a dominating random variable leads to bounds for the asymptotic growth of the AoI process by the results presented in Chapter 5.

6.3.1 AoI moment bounds under α-Mixing communication

To construct uniformly dominating random variables, we derive an upper bound to the complementary cumulative distribution functions (CCDF) $\mathbb{P}(\tau(n) > m)$ uniformly over all $n \geq 0$. Then, we will construct a function $u : \mathbb{N}_0 \to [0, 1]$ such that $\mathbb{P}(\tau(n) > m) \leq u(m)$ for all $m \geq 0$ independent of $n \geq 0$ with $\lim_{m\to\infty} u(m) = 0$. We can now use this bound to define the CCDF of a new random variable. Such a bound can then be used to define a non-negative integer-valued random variable $\overline{\tau}$ by setting

$$\mathbb{P}(\overline{\tau} > m) \coloneqq u(m), \quad \forall m \geq 0. \tag{6.9}$$

This uniquely defines $\overline{\tau}$ and by construction $\overline{\tau}$ stochastically dominates all $\tau(n)$ for all $n \geq 0$. Moreover, if $\sum_{m=0}^{\infty}((m+1)^p - m^p)u(m) < \infty$ for some $p > 0$, then it follows from Proposition 6.4 that $\mathbb{E}[\overline{\tau}^p] < \infty$.

Proposition 6.4 ((Chakraborti, Jardim, and Epprecht 2018))**.** *Suppose X is a non-negative integer-valued random variable, then for every $p > 0$:*

$$\mathbb{E}[X^p] = \sum_{m=0}^{\infty}((m+1)^p - m^p)\mathbb{P}(X > m). \tag{6.10}$$

Violation probability under α-mixing communication

By construction of the AoI process (6.1), we have that

$$\mathbb{P}(\tau(n) > m) = \mathbb{P}\left(\bigcap_{l=n-m}^{n-1} A^c(l)\right), \tag{6.11}$$

where $A^c(n)$ denotes the complement of the event $A(n)$. Observe that if the events $A(n)$ were independent, then the probability on the right-hand side above could directly be written as the product of the individual probabilities. However, we merely consider that $A(n)$ is α-mixing, Assumption 6.2.2. To use this temporal dependency decay, we separate events in (6.11) by intervals of a certain length.

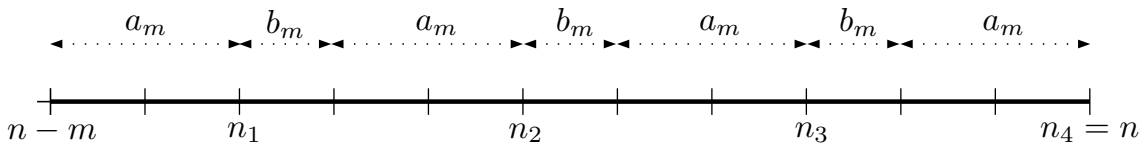

Figure 6.1: Illustration of the time slicing in (6.12).

Let $\{a_m\}$ and $\{b_m\}$ be two non-decreasing sequences of non-negative integers with $a_m \leq m$ and $b_m \leq m$. Now fix $n \geq m \geq 0$ and define time indices

$$n_1 \coloneqq n - m + a_m, \qquad n_k \coloneqq n_{k-1} + a_m + b_m \tag{6.12}$$

as long as $n_k \leq n$. The created time indices are illustrated in Figure 6.1.

Let $L(m)$ be the number of constructed time indices. Notice that removing events from the intersection in (6.11) leads to an upper bound. Thus

$$\mathbb{P}\left(\tau(n) > m\right) = \mathbb{P}\left(\bigcap_{l=n-m}^{n-1} A^c(l)\right) \tag{6.13}$$

$$\leq \mathbb{P}\left(\bigcap_{k=1}^{L(m)} \left(\bigcap_{l=n_k-a_m}^{n_k-1} A^c(l)\right)\right) \tag{6.14}$$

$$= \mathbb{P}\left(\bigcap_{k=1}^{L(m)} \{\tau(n_k) > a_m\}\right). \tag{6.15}$$

Notice that by construction of the time indices n_k, the events $\{\tau(n_k) > a_m\}$ in (6.15) are separated by b_m steps. The following lemma uses this separation to formulate a bound for the joint event in (6.15) using the marginals $\mathbb{P}\left(\tau(n_k) > a_m\right)$ and the mixing coefficients $\alpha(A, n)$ of $A(n)$.

Lemma 6.5. *For $n \geq m \geq 0$,*

$$\mathbb{P}(\tau(n) > m) \leq \prod_{k=1}^{L(m)} \mathbb{P}\left(\tau(n_s) > a_m\right) \alpha(A, b_m) \left(\sum_{k=1}^{L(m)-1} \left(\prod_{s=1}^{k-1} \mathbb{P}\left(\tau(n_s) > a_m\right)\right)\right), \tag{6.16}$$

where n_s is defined in (6.12).

With Assumption 6.2.1, a preliminary bound for the violation probability follows:

Corollary 6.6. *Let $n \geq m \geq 0$, then $\mathbb{P}\left(\tau(n) > m\right) \leq p(m, \delta)$ with*

$$p(m, \delta) \coloneqq \varepsilon^{L_\delta(m)} + \alpha(A, \lceil m^\delta \rceil) \left(\sum_{k=1}^{L_\delta(m)-1} \varepsilon^{k-1}\right) \tag{6.17}$$

for every $\delta \in (0, 1)$ with $L_\delta(m) \coloneqq \left\lfloor \frac{m}{1+\lceil m^\delta \rceil} \right\rfloor$.

Proof. For $\delta \in (0,1)$ choose $a_m = 0$ and $b_m = \lceil m^\delta \rceil$ for all $m \geq 0$ in Lemma 6.5 and use that $\mathbb{P}(\tau(n) > 0) \leq \varepsilon$ for all $n \geq 0$ by Assumption 6.2.1. □

Notice that Lemma 6.5 and Corollary 6.6 hold without Assumption 6.2.2 since the mixing coefficients are defined for all stochastic processes. Under Assumption 6.2.2 it now follows that $\alpha(A,n) \to 0$ as $n \to \infty$. Hence, the right-hand side in (6.17) decays to zero as $m \to \infty$, since $\lim_{m\to\infty} \lceil m^\delta \rceil = \infty$ and $\lim_{m\to\infty} L_\delta(m) = \infty$ for every $\delta \in (0,1)$. The recipe described at the beginning of this section thus immediately shows that a random variable exists that jointly stochastically dominates $\tau(n)$ for all $n \geq 0$. Moreover, for every $q < p$ with p from Assumption 6.2.2, we can choose δ sufficiently close to one such that $p(m,\delta)$ decays sufficiently fast such that

$$\sum_{m=0}^{\infty} ((m+1)^q - m^q) p(m,\delta) < \infty \tag{6.18}$$

Thus for every $q < p$, we can find a random variable $\overline{\tau}$ with $\tau(n) \leq_{\text{st}} \overline{\tau}$ for all $n \geq 0$ and $\mathbb{E}[\overline{\tau}^q] < \infty$.

Proof of Theorem 6.1

The previous paragraph shows Theorem 6.1 for all $q < p$. For $q = p$ the situation is different and the bound in Corollary 6.6 is insufficient to complete Theorem 6.1. This is because b_m in (6.16) has to be chosen such that $\liminf \frac{b_m}{m} > 0$, to guarantee that $\sum_{m=0}^{\infty}((m+1)^p - m^p)\alpha(A,b_m) < \infty$. In this case, $L(m) < \infty$ and thus a_m has to increase to infinity to guarantee that the first term in (6.16) decays to zero. The central observation is that sequences a_m and b_m, as used to construct the time indices in (6.12), have to be chosen to jointly drift to infinity. To do this, we use the bound from Corollary 6.6 in Lemma 6.5 to improve the bound on the violation probability.

Lemma 6.7. *Let $n \geq m \geq 0$,*

$$\begin{aligned} \mathbb{P}(\tau(n) > m) &\leq p(\lceil \lambda m \rceil, \delta)^{L_\lambda(m)} \\ &\quad + \alpha(A, \lceil \lambda m \rceil) \left(\sum_{k=1}^{L_\lambda(m)-1} \left(\prod_{s=1}^{k-1} p(\lceil \lambda m \rceil, \delta) \right) \right) =: u(m,\delta,\lambda) \end{aligned} \tag{6.19}$$

for every $(\delta,\lambda) \in (0,1)^2$ with $L_\lambda(m) := \lfloor \frac{m}{2\lceil \lambda m \rceil} \rfloor$ and $p(m,\delta)$ as defined in Corollary 6.6.

We are now ready to prove Theorem 6.1.

Proof of Theorem 6.1. Fix $(\delta,\lambda) \in (0,1)^2$ and observe that $u(m,\delta,\lambda)$ is by construction decreasing in m. Now define a non-negative integer-valued random variable $\overline{\tau}$ by describing its CCDF as follows:

$$\mathbb{P}(\overline{\tau} > m) := u(m,\delta,\lambda), \quad m \geq 0. \tag{6.20}$$

By Lemma 6.7, $\overline{\tau}$ stochastically dominates all $\tau(n)$ for $n \geq 0$. To prove Theorem 6.1, we have to show that there exist $(\delta, \lambda) \in (0,1)^2$, such that $\sum_{m=0}^{\infty}((m+1)^p - m^p)u(m,\delta,\lambda) < \infty$, which we complete in the appendix. □

6.3.2 Mixing rates and strong law for AoI processes

Section 6.3.1 shows that if $A(n)$ is α-mixing with $\sum_{n=0}^{\infty} n^{p-1}\alpha(A,n) < \infty$, then

$$\mathbb{P}\left(\tau(n) > m\right) \leq \mathbb{P}\left(\overline{\tau} > m\right) \in o(m^{-p}) \tag{6.21}$$

for a random variable $\overline{\tau}$. This uniform tail decay of the distributions of each $\tau(n)$ indicates that the dependency of $\tau(n)$ on $A(m)$ decays as $|n-m| \to \infty$. This, in turn, indicates that the dependency of $\tau(n)$ on $\tau(m)$ also decays. Theorem 6.3 affirms this indication and that $\tau(n)$ is itself α-mixing.

Proof of Theorem 6.3

For $0 \leq l \leq m \leq \infty$, define the σ-algebra generated from $\tau(l)$ up to $\tau(m)$ by

$$\mathcal{F}_l^m(\tau) \coloneqq \sigma\left(\tau(n) \mid l \leq n \leq m\right). \tag{6.22}$$

Similarly, define the σ-algebra generated from $A(l)$ up to $A(m)$ by

$$\mathcal{F}_l^m(A) \coloneqq \sigma\left(A(n) \mid l \leq n \leq m\right). \tag{6.23}$$

The α-mixing coefficients of $\tau(n)$ are

$$\alpha(\tau, n) \coloneqq \sup_{l \geq 0}\{\sup_{A,B}|\mathbb{P}\left(A \cap B\right) - \mathbb{P}\left(A\right)\mathbb{P}\left(B\right)|\}, \tag{6.24}$$

where the supremum is taken over $A \in \mathcal{F}_0^l(\tau), B \in \mathcal{F}_{l+n}^{\infty}(\tau)$.

The idea of Theorem 6.3, is to condition events $A \cap B$ and B in (6.24) on events $\{\tau(l+n) \leq m\}$ and $\{\tau(l+n) > m\}$ for $0 \leq m \leq n$. Then, we use the key property that an event B as above conditioned on $\{\tau(l+n) \leq m\}$ is already an element of $\mathcal{F}_{n+l-m}^{\infty}(A)$, which thus allows us to use that $A(n)$ is α-mixing.

Let us develop some intuition as to why for $B \in \mathcal{F}_{l+n}^{\infty}(\tau)$ it holds that

$$B \cap \{\tau(l+n) \leq m\} \in \mathcal{F}_{l+n-m}^{\infty}(A). \tag{6.25}$$

Observe that information $\tilde{B} \in \sigma(\tau(n+l)) = \mathcal{F}_{n+l}^{n+l}(\tau)$ is of the form

$$\tilde{B} = \tau(n+l)^{-1}(C),$$

where C is a subset of $\{1,\dots,n+l\}$. By definition, we have

$$\{\tau(n+l)\le m\}=\tau(n+1)^{-1}(\{1,\dots,m\}). \tag{6.26}$$

Using the construction of the AoI process (6.1) it is then not difficult to see that $\tilde{B}\cap\{\tau(n+l)\le m\}\in\mathcal{F}^{\infty}_{l+n-m}(A)$. A monotone class argument now completes the reasoning. A formal proof of property (6.25) is given in the chapter appendix. We are now ready to prove Theorem 6.3.

Proof of Theorem 6.3. Let $l,m,n\ge 0$, $A\in\mathcal{F}^{l}_{0}(\tau)$ and $B\in\mathcal{F}^{\infty}_{l+n}(\tau)$. Further, let $\overline{\tau}$ be a dominating random variable as given by Theorem 6.1. By the law of total probability, we have that $|\mathbb{P}(A\cap B)-\mathbb{P}(A)\,\mathbb{P}(B)|$ is equal to

$$\begin{aligned}
&|\mathbb{P}(A\cap B\cap\{\tau(l+n)\le m\})+\mathbb{P}(A\cap B\cap\{\tau(l+n)>m\})\\
&\quad-\big(\mathbb{P}(A)\,\mathbb{P}(B\cap\{\tau(l+n)\le m\})+\mathbb{P}(A)\,\mathbb{P}(B\cap\{\tau(l+n)>m\})\big)| &(6.27)\\
&\le|\mathbb{P}(A\cap B\cap\{\tau(l+n)\le m\})-\mathbb{P}(A)\,\mathbb{P}(B\cap\{\tau(l+n)\le m\})|\\
&\quad+\mathbb{P}(\tau(l+n)>m)\Big(|\mathbb{P}(A\cap B\mid\{\tau(l+n)>m\})-\mathbb{P}(A)\,\mathbb{P}(B\mid\{\tau(l+n)>m\})|\Big) &(6.28)\\
&\le|\mathbb{P}(A\cap B\cap\{\tau(l+n)\le m\})-\mathbb{P}(A)\,\mathbb{P}(B\cap\{\tau(l+n)\le m\})|+\mathbb{P}(\overline{\tau}>m) &(6.29)
\end{aligned}$$

The first inequality uses conditional probability and triangular inequality; the second inequality uses that $\tau(n)\le_{\mathrm{st}}\overline{\tau}$ for all $n\ge 0$ and that

$$|\mathbb{P}(A\cap B\mid\{\tau(l+n)>m\})-\mathbb{P}(A)\,\mathbb{P}(B\mid\{\tau(l+n)>m\})|\le 1. \tag{6.30}$$

By construction of the AoI process, we have that $\mathcal{F}^{l}_{0}(\tau)\subset\mathcal{F}^{l}_{0}(A)$ for all $l\ge 0$. Thus $A\in\mathcal{F}^{l}_{0}(A)$. By (6.25), we have that $B\cap\{\tau(l+n)\le m\}\in\mathcal{F}^{\infty}_{l+n-m}(A)$. Since $A(n)$ is α-mixing it follows that

$$|\mathbb{P}(A\cap B\cap\{\tau(l+n)\le m\})-\mathbb{P}(A)\,\mathbb{P}(B\cap\{\tau(l+n)\le m\})|\le\alpha(A,n-M) \tag{6.31}$$

Thus for all $n\ge 0$, we found that

$$\alpha(\tau,n)\le\alpha(A,n-m)+\mathbb{P}(\overline{\tau}>m) \tag{6.32}$$

for all $0\le m\le n$. To see that $\alpha(\tau,n)\to 0$ as $n\to\infty$ choose, e.g., $m(n)=\lceil\frac{n}{2}\rceil$. Minimizing over m in (6.32) yields the statement of Theorem 6.3. □

Mixing rate of $\tau(n)$

The next natural question is to analyze the convergence rate of $\alpha(\tau,n)$. For this, let $q<p$ and let $\delta\in(0,1)$. Then

$$\sum_{n=0}^{\infty}n^{q-1}\alpha(\tau,n)\le\sum_{n=0}^{\infty}n^{q-1}\alpha(A,\lceil n^{\delta}\rceil)+\sum_{n=0}^{\infty}n^{q-1}\mathbb{P}\left(\overline{\tau}>n-\lceil n^{\delta}\rceil\right). \tag{6.33}$$

By Assumption 6.2.2, we have that $\sum_{n=0}^{\infty} n^{p-1}\alpha(A,n)$ and we can show using $q<p$ that the first summation is finite *for some* $\delta \in (0,1)$. We also claim that the second summation is finite *for every* $\delta \in (0,1)$. This follows from (6.21) and the observation that $\sum_{n=0}^{\infty} n^{q-1-p} < \infty$ for $q<p$. Here, we used that $\lim_{n\to\infty} \frac{n^p}{(n-\lceil n^\delta \rceil)^p} = 1$ for every $\delta \in (0,1)$. We have, therefore, shown that

$$\sum_{n=0}^{\infty} n^{p-1}\alpha(A,n) < \infty \implies \sum_{n=0}^{\infty} n^{q-1}\alpha(\tau,n) \quad \forall q<p. \tag{6.34}$$

Thus, $\tau(n)$ has almost the same mixing rate as $A(n)$.

As a corollary to Theorem 6.3 and the mixing rate in (6.34), we can now formulate a SLLN for $\tau(n)$. The SLLN is based on an SLLN for strongly mixing stochastic processes that we state here paraphrased to suit the purpose:

Theorem 6.8 ((McLeish 1975, Thm. 2.10))**.** *Suppose* $\{X_n\}_{n\geq 0}$ *is a zero mean* α*-mixing sequence of random variable with* $\sum_{n=0}^{\infty} \alpha(X,n)^{\frac{1}{q}} < \infty$ *for some* $q>1$*. If there is* $1<r<q$ *and* $\frac{r}{r-1} < s \leq \frac{2r}{r-1}$ *such that*

$$\sum_{n=0}^{\infty} n^{-\frac{r-1}{r}s} \mathbb{E}\left[|X_n|^{\frac{r-1}{r}s}\right] < \infty, \tag{6.35}$$

then $\frac{1}{N}\sum_{n=0}^{N-1} X_n \to 0$ *a.s.*

Using the moment bound in Theorem 6.1 and the mixing rate in Theorem 6.3, we can carefully choose r,s in Theorem 6.8 to conclude with the following SLLN for $\tau(n)$.

Corollary 6.9. *Suppose the assumptions of Theorem 6.1 hold for some* $p>1$*, then*

$$\frac{1}{N}\sum_{n=0}^{N-1}(\tau(n) - \mathbb{E}\left[\tau(n)\right]) \xrightarrow{a.s.} 0. \tag{6.36}$$

If in addition $A(n)$ *is stationary, then*

$$\frac{1}{N}\sum_{n=0}^{N-1}\tau(n) \xrightarrow{a.s.} \lim_{n\to\infty} \mathbb{E}\left[\tau(n)\right] \leq \mathbb{E}\left[\overline{\tau}\right] \tag{6.37}$$

6.4 Discussion and related work

AoI has been studied for various queuing and scheduling models (Yin Sun et al. 2019) that require i.i.d. service and waiting times as in renewal theory. The most common analyses rely on the saw-tooth nature of AoI processes (see Figure 1.3) in combination with certain stationarity and ergodicity assumptions for the waiting and service time processes (Yates et al. 2021). In wireless communication settings, AoI has been considered in edge-based network models where the success of communication via individual edges has been considered as i.i.d. across time (Farazi,

Klein, and Brown 2020). Generally, we observed that i.i.d. assumptions are prominent for arrival/service times and communication success. In the former case, this is due to the tractability of sums of independent random variables using laws of large numbers. In the latter case, it is due to the tractability of products of independent random variables. However, interarrival times can be highly dependent as arrivals tend to form clumps (D. Daley and Rolski 1992). Similarly, communication over wireless fading channels can be highly correlated (G. Bianchi 2000; Boban, Gong, and W. Xu 2016; W. Liu and Shi 2019). Independence assumptions are, therefore, particularly unrealistic for communication systems. In contrast, a fundamental property of communication systems is that events that occur "close" in some domains (space, time, frequency, or code) are highly dependent. Still, the dependency decreases as the events get more "separated". In general, many real-world systems possess potentially "long-range" dependencies that decay as the events get more separated in time (Samorodnitsky et al. 2016).

The lack of literature that considered non-independent service times motivated the consideration of the fundamental AoI process (6.1) as a function of an event process $A(n)$ with dependencies described by a suitably probabilistic mixing notion. Previously, a version of the AoI process (6.1) has only been studied under the assumption that the event process $A(n)$ is an independent stochastic process (Montero and Villarroel 2016). This work takes the perspective of the AoI process (6.1) as a random walk with restarts, called Sisyphus random walk, due to its analogy with climbing and falling down a hill indefinitely. The work analyzes (6.1) as a countable-state Markov process. Hence, before the results presented herein, the AoI process (6.1) was not studied when $A(n)$ is a general (not necessarily stationary) stochastic process with time dependencies.

The presented analysis of the abstract AoI process (6.1) under merely α-mixing communication is a gateway to study more complex AoI processes with dependent communication. For example, renewal processes with non-independent interarrivals or real-world AoI processes where the mixing rate of the communication process has been estimated from data. The required α-mixing assumption, Assumption 6.2.2, requires that the dependency of events $A(n)$ and $A(m)$ decays as $|m-n| \to \infty$. There are many examples where $A(n)$ will be α-mixing. In general, $A(n)$ will be α-mixing if it can be written as a Borel-measurable function of another α-mixing process (Bradley 2005, Thm. 5.2). Here, an important class of examples are scenarios where the communication events $A(n)$ are a Borel-measurable function of a geometrically ergodic Markov process (Davydov 1974; Bradley 2005), which has been a common tool to approximate wireless fading channels (H. S. Wang and Moayeri 1995; G. Bianchi 2000; Boban, Gong, and W. Xu 2016). We studied this class in (Redder, Ramaswamy, and Karl 2022e), where an event process that represents successful communication via a geometrically ergodic wireless fading channel was shown to be α-mixing even when AoI-aware medium access control protocols are used to decide when to communicate. The results will be discussed in Chapter 8.

Another line of applications comes from point process theory. Berbee (1987, Theorem 6.1) shows that the event process resulting from a lattice renewal process with i.i.d. interarrival times is α-mixing if the waiting time distribution has a finite moment greater than one. Pham and Tran (1985) determined the α-mixing rate of linear processes and ARMA processes. Notably, general point processes with temporally dependent interarrivals, where a probabilistic mixing notion describes the dependency, have not been considered so far. The presented results offer a path to analyze AoI in such settings even when new information is received at a monitor after general dependent interarrival times. To apply the results, we need to use relation (1.11) between the backward recurrence time of a point process and the associated fundamental AoI process, which shows that

$$\tau_r(n) = \tau(n - \tau(n)) + \tau(n) - 1 \tag{6.38}$$

holds, where $\tau_r(n)$ is the backward recurrence time and $\tau(n)$ is the fundamental AoI process defined by (6.1) with $A(n)$ the event process tracking the occurrence of points. Property (6.38) is proven below in the chapter appendix. To complete the application, we need to compute the α-mixing rate of the event sequence when the sequence of dependent interarrival times is itself α-mixing. As of recently, the α-mixing rates of renewal sequences were only known (asymptotically) when the associated interarrival times are i.i.d. Berbee (1987, Theorem 6.1).

Another interesting line of work will be identifying scenarios where modeling the process that gives rise to AoI is difficult. For such scenarios, we envision that α-mixing rates of the event process $A(n)$ can be directly estimated from data using the estimator of Khaleghi and Lugosi (2023). Then, the results presented in this chapter enable conclusions about the average AoI using Theorem 6.3.

6.5 Proofs of Chapter 6

Proof of Lemma 6.5. We expand the right-hand side of (6.15) using Assumption 6.2.2. Consider the following σ-algebras:

$$\mathcal{F}_l^s \coloneqq \sigma\left(A(n) \mid l \leq n \leq s\right), \quad l \geq 0, s \geq 0. \tag{6.39}$$

Each event $\{\tau(n_k) > a_m\}$ in (6.15) is generated by the events $A(n)$ with $n \in \{n_k - a_m, \ldots, n_k - 1, n_k - 1\}$. By construction of the above sub-σ-algebras, we have that

$$\{\tau(n_k) > a_m\} \in \mathcal{F}_{n_k - a_m}^{n_k - 1}. \tag{6.40}$$

Recall that by the construction of the time indices (6.12), the events $\{\tau(n_k) > a_m\}$ are separated by b_m steps. We thus have that

$$\{\tau(n_{L(m)}) > a_m\} \in \mathcal{F}_{n_{L(m)} - a_m}^{\infty} \tag{6.41}$$

and

$$\bigcap_{k=1}^{L(m)-1} \{\tau(n_k) > l_m\} \in \mathcal{F}_0^{n_{L(m)-1}-1}, \tag{6.42}$$

where $L(m)$ is the number of constructed time indices. Due to the aforementioned separation, we have

$$n_{L(m)} - a_m - (n_{L(m)-1} - 1) = b_m + 1 \geq b_m. \tag{6.43}$$

By Assumption 6.2.2 $\{A(n)\}_{n\geq 0}$ is α-mixing with coefficient $\alpha(A, n)$. It then follows from (6.41) and (6.42) and the construction of the time indices n_k that

$$\mathbb{P}\left(\tau(n) > m\right) \leq \mathbb{P}\left(\tau(n_{L(m)}) > a_m\right) \mathbb{P}\left(\bigcap_{k=1}^{L(m)-1} \{\tau(n_k) > a_m\}\right) + \alpha(A, b_m). \tag{6.44}$$

The lemma then follows by applying the described procedure successively. □

Proof of Lemma 6.7. For $\lambda \in (0, 1)$ choose $a_m = b_m = \lceil \lambda m \rceil$ for all $m \geq 0$ in Lemma 6.5. Then, (6.17) shows that for every $\delta \in (0, 1)$ and every $n \geq 0$, we have the bound

$$\mathbb{P}\left(\tau(n) > a_m\right) \leq p(\lceil \lambda m \rceil, \delta). \tag{6.45}$$

□

Proof of Theorem 6.1. Fix $(\delta, \lambda) \in (0, 1)^2$ and observe that $u(m, \delta, \lambda)$ is by construction decreasing in m. Now define a non-negative integer-valued random variable $\overline{\tau}$ by describing its CCDF as follows:

$$\mathbb{P}\left(\overline{\tau} > m\right) := u(m, \delta, \lambda), \quad m \geq 0. \tag{6.46}$$

By Lemma 6.7, $\overline{\tau}$ stochastically dominates all $\tau(n)$ for $n \geq 0$. To prove Theorem 6.1, we have to show that there exist $(\delta, \lambda) \in (0, 1)^2$, such that $\sum_{m=0}^{\infty}((m+1)^p - m^p)u(m, \delta, \lambda) < \infty$.

We start by showing that

$$\sum_{m=0}^{\infty}((m+1)^p - m^p)\alpha(A, \lceil \lambda m \rceil) < \infty \tag{6.47}$$

for all $\lambda \in (0, 1)$. We claim that

$$\sum_{m=1}^{\infty}((m+1)^p - m^p)\alpha(\lceil \lambda m \rceil) \leq \frac{2^p}{\lambda^p} \sum_{m=1}^{\infty} m^{p-1}\alpha(m) \tag{6.48}$$

This claim follows from the observations that for $x \in \mathbb{R}_{\geq 0}$ and $m \geq 1$, $((m+1)^x - m^x) \leq 2^x m^{x-1}$, and that $|\{n \geq 0 : \lceil \lambda n \rceil = m\}| \leq \frac{1}{\lambda}$. Hence, by Assumption 6.2.2 we have that (6.47) holds for all $\lambda \in (0, 1)$. Further, $L_\lambda(m) \leq \frac{1}{\lambda}$. To complete the proof it is therefore left to show that

$$\sum_{m=0}^{\infty}((m+1)^p - m^p)p(\lceil \lambda m \rceil, \delta)^{L_\lambda(m)} < \infty \tag{6.49}$$

for some $(\delta, \lambda) \in (0,1)^2$. Since $(m+1)^p - m^p \leq 2^p m^{p-1}$ for $m \geq 1$, its enough to show that $\sum_{m=1}^{\infty} m^{p-1} p(\lceil \lambda m \rceil, \delta)^{L_\lambda(m)} < \infty$. Using the summability property of $\alpha(A,n)$ from Assumption 6.2.2, we can show that

$$p(\lceil \lambda m \rceil, \delta)^{L_\lambda(m)} \in \mathcal{O}\left(\left(\varepsilon^{\left(m^{1-\delta}\right)} + m^{-\mu\delta}\right)^{\frac{1}{\lambda}}\right) \tag{6.50}$$

for $\mu \coloneqq p$, if $p \leq 1$ and $\mu \coloneqq p-1$, if $p > 1$.[1] Asymptotically, $m^{-\mu\delta}$ will dominate $\varepsilon^{\left(m^{1-\delta}\right)}$ for any $\delta \in (0,1)$. It is thus enough to show that $\sum_{m=1}^{\infty} m^{p-1} m^{-\mu\frac{\delta}{\lambda}} < \infty$, which holds for $\delta\mu > \lambda p$. This completes the proof. □

Proof of missing property. We verify that for $B \in \mathcal{F}_{l+n}^{\infty}(\tau)$, we have that

$$B \cap \{\tau(l+n) \leq M\} \in \mathcal{F}_{l+n-M}^{\infty}(A), \tag{6.51}$$

as used in Section 6.3.2.

First consider $B \in \sigma(\tau(n+l))$. Every AoI random variable is a measurable map $\tau(n+l) : \Omega \to 2^{\{1,\dots,n+l\}}$. Thus B is of the form $B = \tau(n+l)^{-1}(C)$ for some $C \in 2^{\{1,\dots,n+l\}}$, i.e. C is a subset of $\{1,\dots,n+l\}$. Second, for any $0 \leq m \leq n+l$, we have

$$\{\tau(n+l) \leq m\} = \tau(n+1)^{-1}(\{1,\dots,m\}). \tag{6.52}$$

Therefore, using properties of the preimage of intersections, we have that

$$B \cap \{\tau(n+l) \leq m\} = \tau(n+l)^{-1}\left(C \cap \{1,\dots,m\}\right) \tag{6.53}$$

$$= \bigcup_{c \in C \cap \{1,\dots M\}} \tau(n+l)^{-1}(\{c\}). \tag{6.54}$$

By construction of the AoI process, we have that

$$\tau(n+l)^{-1}(c) = A(n+l-c) \cap \bigcap_{k=1}^{c-1} A^c(n+l-c+k). \tag{6.55}$$

Since $\mathcal{F}(A)_{l+n-m}^{\infty}$ is a σ-algebra, we thus conclude from (6.54) and (6.55) that

$$B \cap \{\tau(n+l) \leq m\} \in \mathcal{F}(A)_{l+n-m}^{\infty}. \tag{6.56}$$

Next we consider elements of the join σ-algebra $\mathcal{F}(\tau)_{n+l}^{\infty}$. It can be expressed as

$$\mathcal{F}(\tau)_{n+l}^{\infty} = \sigma\left(\bigcup_{k=n+l}^{\infty} \sigma(\tau(k))\right). \tag{6.57}$$

For all $B \in \bigcup_{k=n+l}^{\infty} \sigma(\tau(k))$, the previous paragraph shows that

$$B \cap \{\tau(n+l) \leq m\} \in \mathcal{F}(A)_{l+n-m}^{\infty}, \tag{6.58}$$

[1] The distinction is necessary, since $m^{p-1}\alpha(A,n)$ is not necessarily monotone for $p > 1$.

using the stability of σ-algebras under countable unions. A generating-class argument now completes the proof: Let $\mathcal{G} \coloneqq \{B \in \mathcal{F}(\tau)_{n+l}^{\infty} : B \text{ satisfies } (6.56)\}$. Clearly, $\Omega \in \mathcal{G}$ and countable unions of elements from $\mathcal{G}$ are in $\mathcal{G}$. Finally, let $B \in G$. Then

$$B^c \cup \{\tau(n+l) > m\} \in \mathcal{F}(A)_{l+n-m}^{\infty}, \tag{6.59}$$

since $B \cap \{\tau(n+l) \leq m\} \in \mathcal{F}(A)_{l+n-m}^{\infty}$ and $\mathcal{F}(A)_{l+n-m}^{\infty}$ is a σ-algebra. Finally,

$$\begin{aligned} B^c \cap \{\tau(n+l) \leq m\} &= (B^c \cup \{\tau(n+l) > m\}) \\ &\cap \{\tau(n+l) \leq m\} \in \mathcal{F}(\tau)_{n+l-m}^{\infty}, \end{aligned} \tag{6.60}$$

again, since $\{\tau(n+l) \leq m\} \in \mathcal{F}(\tau)_{n+l-m}^{\infty}$ and $\mathcal{F}(A)_{l+n-m}^{\infty}$ is a σ-algebra. We have therefore shown that $\mathcal{G}$ is itself a σ-algebra, hence $\mathcal{G} = \mathcal{F}(\tau)_{n+l-m}^{\infty}$. □

Proof of Corollary 6.9. Suppose that $A(n)$ is α-mixing with $\sum_{n=0}^{\infty} n^{p-1}\alpha(A,n) < \infty$ for some $p > 1$. Theorem 6.3 shows that $\tau(n)$ is α-mixing with $\sum_{n=0}^{\infty} n^{q-1}\alpha(\tau,n) < \infty$ for every $q < p$. Let $1 < q < p$. To apply Theorem 6.8 to $\tau(n)$, we first have to show that

$$\sum_{n=0}^{\infty} \alpha(\tau,n)^{\frac{1}{q}} = \sum_{n=0}^{\infty} \alpha(\tau,n)^{\frac{1}{q}-1}\alpha(\tau,n) < \infty \tag{6.61}$$

Since $\sum_{n=0}^{\infty} n^{q-1}\alpha(\tau,n) < \infty$ and $\alpha(\tau,n)$ is monotone, we especially have that $\alpha(\tau,n) \in o(n^{-1})$ and hence $\alpha(\tau,n)^{\frac{1}{q}-1} \in o(n^{1-\frac{1}{q}})$. Finally (6.61) follows, since $1 - \frac{1}{q} \leq q - 1$ for $q \geq 0$ and $\sum_{n=0}^{\infty} m^{q-1}\alpha(\tau,n) < \infty$ by Assumption 6.2.2.

To complete the proof, recall that Theorem 6.1 showed that $\mathbb{E}[\tau(n)^p] \leq \mathbb{E}[\overline{\tau}^p] < \infty$. It is now easy to see that we can choose r, s in Theorem 6.8, such that $1 < \frac{r-1}{r}s < p$ and thus (6.35) holds. Theorem 6.8 therefore shows that $\frac{1}{N}\sum_{n=0}^{N-1}(\tau(n) - \mathbb{E}[\tau(n)]) \xrightarrow{\text{a.s.}} 0$. If $A(n)$ is strictly stationary, then $\mathbb{E}[\tau(n)]$ is monotonically increasing and the additional statement follows. □

Proof of Equation (1.11). We drop the node indices for this proof. Fix some $n \geq 0$. First, if $A(n) = 1$, then $\tau_r(n) = W_{k(n)}$. Further, $\tau(n) = 1$ and thus $\tau(n-\tau(n)) = \tau(n-1) = 1 + (W_{k(n)} - 1) = W_{k(n)}$, since $\tilde{\tau}(n-1)$ is reset to 1 at the next time step. Hence, $\tau(n-\tau(n)) + \tau(n) - 1 = W_{k(n)}$ holds as desired. Second, if $A(n) = 0$, then $\tau_r(n) = n - \sum_{i=1}^{k(n)-1} W_i$, as the age is the time that was spent waiting since the last renewal plus the associated last transmission time. Further, $\tau(n) = n - \sum_{i=1}^{k(n)} W_i + 1$ and thus

$$\tau(n - \tau(n)) = \tau\Big(\sum_{i=1}^{k(n)} W_i - 1\Big) = W_{k(n)}, \tag{6.62}$$

since $\sum_{i=1}^{k(n)} W_i$ is a renewal time step. Again, $\tau(n - \tau(n)) + \tau(n) - 1 = W_{k(n)}$ holds as desired.

□

Chapter 7

AoI from asynchronous computing modeled as parallel renewal processes

In the last chapter, we saw how event processes give rise to AoI processes. Such AoI processes are fundamental to describing the AoI between a single source and a single monitor. We now consider the asynchronous computing setting described in Section 1.3.2. Specifically, this chapter studies AoI arising from asynchronous computing modeled as parallel point processes. In addition, we will explain how the results of the previous chapter can be used inside this distributed computing model. The results are based on (Redder 2023) and provide the core answer to (Q8) on the distribution of AoI caused by asynchronous computing. Furthermore, the results guarantee the existence of dominating random variables with precise moment bounds to apply the results of Chapter 5 and answer (Q4).

7.1 Asychronous computing models

We consider two asynchronous parameter server systems, where workers sequentially read the parameters of a model and a minibatch from a data set and then asynchronously compute optimization steps to update the model. As explained in Section 1.2 and Section 1.3.2, this leads to increased resource utilization at the cost of parameter update/staleness errors due to AoI, which mitigate the performance of iterative methods when inappropriate hyperparameters are chosen. Because of this, we study how processing times and asynchronous computing on distributed computing (DC) systems give rise to AoI to design methods that maximize resource utilization while guaranteeing performance. Specifically, we propose to model a DC system with parallel workers by parallel renewal processes describing the workers' processing times. For this model, we derive exact asymptotic AoI distributions and AoI moment bounds affecting a class of asynchronous parameter server algorithms. The results then enable the allocation of computing

resources for asynchronous methods based on processing time data readily available from DC system traces (Samsi et al. 2021), which we will be exemplified in Section 7.5. Finally, we will discuss that the results naturally generalize to arbitrary point processes with weakly convergent increments.

7.1.1 AoI caused by asynchronous parameter updates

We now recall the AoI caused by asynchronous methods executed on parallel computing infrastructure. These methods fall into a general class of asynchronous parameter server iterations (APSI) and coordinate-wise parameter server iterations (cAPSI), formally stated as Algorithm 2 and Algorithm 3 in Section 7.2.1, respectively. In Section 1.3.2, we explained how update steps that are returned to a single parameter server (an APSI) have already aged when applied due to the updates of other workers. Precisely, the resulting API is $\tau(n) = n - m(n)$, where $m(n)$ is the index to which an update step applied at index n corresponds. Then $\tau(n) \coloneqq n - m(n)$ and we refer to $\{\tau(n)\}$ as the sequence of AoI random variables for an APSI. An illustration for an AoI sequence generated by ASGD with two workers was presented in Figure 1.5.

The cAPSI model represents asynchronous learning methods where individual workers update parts of a large neural network model independently (Guan et al. 2019). Most importantly, as AI and deep learning models grow, complete update steps can not be computed anymore on single chips, so coordinate-wise updates become necessary (B. Yang et al. 2021). The cAPSI model captures this. cAPSI generally encompasses physically distributed asynchronous learning and optimization scenarios, e.g., as multi-agent reinforcement learning discussed in Chapter 4. In addition, the cAPSI model can arise due to inconsistent read-and-write problems in APSI (Lian et al. 2015). Finally, both APSI and cAPSI apply to classical distributed and federated learning scenarios (C. Zhang et al. 2021) since APSI and cAPSI will not differentiate between the actual information sent to the parameter servers. For federated learning, the AoI will then be associated with the update information from one worker/client and its local data set.

For cAPSI, the parameter space is divided into subspaces, e.g., $\mathbb{R}^d = \mathbb{R}^{d_1} \times \ldots \times \mathbb{R}^{d_D}$, such that $\sum_i d_i = d$, where d would be the parameter space dimension for APSI. All D coordinates of cAPSI are then updated independently such that D^2 AoI sequences $\tau_{ij}(n)$, $1 \leq i, j, \leq D$, arise. These AoI sequences are defined analogously to the APSI AoI sequences. cAPSI with a single coordinate thus reduces to APSI, but it is simpler and illustrative to first study APSI. For APSI and cAPSI, the updates' processing times will be modeled as parallel renewal processes. As mentioned before, the crucial factor is that events are not pooled, but instead, the object of interest is the number of events (parameter updates) that occur from other renewal processes while one renewal process is waiting for its next event to occur.

7.1.2 Illustrative results

For APSI and cAPSI, we develop closed-form expressions for the limiting AoI distribution from the processing time distributions of the DC system workers, see Theorem 7.2 and Theorem 7.11, respectively. For the limiting mean AoI, we have the following illustrative results:

- Corollary 7.5: For APSI with K workers and heterogeneous but independent processing times, the AoI satisfies:

$$\lim_{n\to\infty} \mathbb{E}\left[\tau(n)\right] = K - 1. \tag{7.1}$$

- Corollary 7.12: For cAPSI with K_i workers on each coordinate (and heterogeneous but independent processing times), the AoI sequences satisfy:

$$\lim_{n\to\infty} \mathbb{E}\left[\tau_{ij}(n)\right] = \frac{\sum_{k=1}^{K_i} \frac{1}{\mu_{ik}}}{\sum_{k=1}^{K_j} \frac{1}{\mu_{jk}}} \times \begin{cases} K_j & i \neq j, \\ K_j - 1 & i = j, \end{cases} \tag{7.2}$$

  where μ_{ik} is the mean processing time of the k-th worker on the i-th coordinate.

Corollary 7.5 shows that independent of the actual processing time distributions, the limiting mean AoI is given by $K-1$. This result reinforces recent observations made for ASGD, which uses that "most" AoI affecting the iteration are actually in $\mathcal{O}(K)$ (Mishchenko et al. 2022; Anastasiia Koloskova, Stich, and Jaggi 2022). The result becomes less surprising with a simple example. Consider $K = 2$ workers with constant processing times, where worker 1 does 10^6 updates while worker 2 only computes a single update. Then, for the initial 10^6 updates, the AoI is zero, while for the update by worker 2, the AoI is 10^6, and $\lim_{n\to\infty} \mathbb{E}\left[\tau(n)\right] = \frac{(10^6-1)\times 0 + 1\times 1 + 1\times 10^6}{10^6+1} = 1 = K - 1$. In other words, the straggler has no effect. Slow workers will affect the overall processing time, but from the perspective of information delay affecting ASGD, the actual processing time distributions do not affect the mean AoI asymptotically; only the number of workers matters. This is a crucial property for the design of asynchronous optimization algorithms. *In view of Chapter 3, ASGD with a single asynchronous update iteration will, therefore, always have AoI with bounded first moment independent of the processing times.* However, as we will see in Section 7.3, the processing time distributions affect higher-order AoI moments.

For cAPSI, the situation is different. For coordinate-wise updates with $K_1 = K_2 = 1$ with the workers from the APSI example, the limiting AoI mean satisfies $\lim_{n\to\infty} \mathbb{E}\left[\tau_{12}(n)\right] = 10^6$ and $\lim_{n\to\infty} \mathbb{E}\left[\tau_{21}(n)\right] = \frac{1}{10^6}$. This scenario is artificial but highlights that worker scheduling is essential for cAPSI. More precisely, equation (7.2) shows that a newly added "fast" worker on one coordinate will increase the associated mean AoI for all coordinates and will thus negatively affect the whole cAPSI iteration. As mentioned before, the cAPSI model especially applies to asynchronous learning methods where individual workers update parts of a large neural network model independently. Based on AoI moment bounds like (7.2), we can then design resource allo-

cation problems to optimize AoI properties affecting asynchronous methods and thus maximize their convergence rate. This is exemplified in Section 7.5.

7.2 System Models

We abstract the DC system model by stochastic processing times completed at time instances of renewal processes as defined in Definition 0.0.10. By assuming that renewal processes describe the processing times, we can precisely characterize the discrete AoI affecting parameter iteration due to asynchronous computing. Notably, the processing times may include the time to send a job, the waiting time of the job in the worker queue, the computing time for the job, potential additional waiting time due to preemption, the time to send the update back, the time to apply the computed update, etc.

The following example illustrates how stochastic processing times naturally arise for sequential tasks that compete with other jobs for service time on a DC system.

Example 7.2.1. *Consider the setting illustrated in Figure 1.1: A stream of jobs arrives at a DC system, and a scheduler assigns jobs to the workers. Scheduled jobs are added to the workers' queue and are processed according to a local scheduling policy, e.g., FCFS, generalized processor sharing, etc. In addition, a highly parallelized iterative algorithm sequentially adds parameter update jobs to the queues of the workers. In other words, the algorithm competes with the arriving (competing) jobs for service time on the processors. In this sense, the iterative algorithm runs concurrently with other jobs on the DC system. Due to the non-trivial effect of priorities and scheduling, the parameter updates will experience stochastic processing times that can be modeled as renewal processes or point processes*

The next two subsections complete the description of the APSI and cAPSI implementations and their associated AoI processes.

7.2.1 AoI for asynchronous parameter server iterations (APSI)

For APSI, we assume that the DC system is modeled as K parallel renewal process:

Assumption 7.2.1. *For each worker k, the processing times for the completion of parameter update jobs are given by an i.i.d. sequence of non-negative real numbers $\{W_k(n)\}$ with $\mu_k \coloneqq \mathbb{E}[W_k(n)] < \infty$.*

In the following, we use W_k to denote a random variable with $W_k \sim W_k(n), n \geq 1$. APSI considers a single parameter iteration maintained by a server. The generic APSI is presented in Algorithm 2.

Algorithm 2 Generic Asynchronous Parameter Server Iteration (APSI)

Initialize the parameter iteration $x_0 \in \mathbb{R}^d$.
for the entire duration **do**
    for each worker **do**
        Receive parameter update job.
        Complete other jobs in the queue.
        Compute the parameter update.
        Send the update to the iteration.
    for the parameter iteration **do**
        Receive an update step from a worker.
        Update the parameter iteration.
        Sent a new job to the worker.

The index n is the counter of the parameter iteration. Using this index, we define AoI random variables associated with each worker. For every worker $k \in \{1, \dots, K\}$, let $x_{m_k(n)}$ be the parameter-iterate in the most recent update job assigned to worker k while the master iterate is at step n. Then define the *worker AoI* sequence $\tau_k(n) \coloneqq n - m_k(n)$. In other words, $\tau_k(n)$ tracks the AoI of the update to be computed by worker k. Consequently, $\tau(n) = \tau_k(n)$ if worker k contributes the update step to update the parameter iteration from x_n to x_{n+1}. The key observation is that the AoI sequence $\tau(n)$ always takes values of the worker AoI sequences $\tau_k(n)$, in other words Definition 5.1.2 applies. We now derive expressions for all $\tau_k(n)$.

Define for $n \geq 1$ the sum of the first n processing times of worker k as

$$S_k(n) \coloneqq \sum_{i=1}^{n} W_k(i). \tag{7.3}$$

Each $S_k(n)$ is thus the n-th renewal time of the k-th worker. Further, define for each $t \geq 0$ and worker k the number of renewals until time t:

$$N_k(t) \coloneqq \max\{n : S_k(n) \leq t\}. \tag{7.4}$$

For any time $t \geq 0$ and any worker k, the worker AoI in continuous time can now be stated as

$$\tilde{\tau}_k(t) \coloneqq \sum_{j \neq k} \left(N_j(t) - N_j(S_k(N_k(t)))\right), \tag{7.5}$$

$$= \sum_{j \neq k} \left(N_j(t) - N_j(t - B_k(t))\right), \tag{7.6}$$

with $B_k(t) \coloneqq t - S_k(N_k(t))$, i.e., $B_k(t)$ is the backward recurrence time (Definition 0.0.12) at time t. Next, consider for every $n \geq 0$ the associated continuous time $t(n)$, where the n-th parameter update occurred. Formerly, this time can be defined as

$$t(n) \coloneqq \max_{n_1 + \dots + n_K = n} S_k(n_k). \tag{7.7}$$

Notice that each sample path has a unique tuple $(n_1, \ldots, n_K)$, such as $n_1 + \ldots + n_K = n$. The connection between the discrete-time worker AoI $\tau_k(n)$ and the continuous-time worker AoI $\tau_k(t)$ is thus given by

$$\tau_k(n) = \tilde{\tau}_k(t(n)_-), \tag{7.8}$$

where $\tilde{\tau}_k(t(n)_-) \coloneqq \lim_{t \nearrow t(n)} \tilde{\tau}_k(t)$ denotes the left limit. The left limit of $\tilde{\tau}_k(t)$ makes precise that $\tau_k(n)$ takes the value of the continuous time worker AoI sequence at the instance when the worker completes its update. We can now state the AoI sequence of APSI as follows:

$$\tau(n) = \sum_{k=1}^{K} \mathbb{1}_{\{t(n)=S_k(N_k(t(n)))\}} \tilde{\tau}_k(t(n)_-). \tag{7.9}$$

Recall that $\mathbb{1}_E$ denotes the indicator function of an event $E \in \mathcal{F}$. The expression formalizes the observation above that $\tau(n)$ always takes values of one of the worker AoI sequences $\tau_k(n)$. By construction, (7.9) fits the definition of an AoI process presented in Chapter 5.

7.2.2 AoI for coordinate-wise APSI

The further distributed iteration cAPSI considers $D \in \mathbb{N}$ coupled, parallel parameter iterations maintained by one or many servers. Let K_i be the number of workers that compute update for coordinate iteration $i \in \{1, \ldots, D\}$. As for APSI, we assume renewal processes describe the workers' processing times.

Assumption 7.2.2. *For the k-th worker on coordinate $i \in \{1, \ldots, D\}$, the processing times for the completion of parameter update jobs are given by an i.i.d. sequence of non-negative real numbers $\{W_{ik}(n)\}_{n \geq 1}$ with $\mu_{ik} \coloneqq \mathbb{E}[W_{ik}(n)] < \infty$.*

The generic cAPSI is presented in Algorithm 2. The main difference in Algorithm 3 compared to Algorithm 2 is the presence of multiple iterations updated asynchronously. Further, there is an operation to collect the current parameters from the other iterations. This step could alternatively be executed by the workers.

The random variables (7.3)-(7.9) defined in Section 7.2.1 can now be defined analogously for cAPSI. For each pair (i, j), $1 \leq i, j \leq D$, the AoI sequences is thus given by

$$\tau_{ij}(n) = \sum_{k=1}^{K_j} \mathbb{1}_{\{t_j(n)=S_{jk}(N_{jk}(t_j(n)))\}} \tilde{\tau}_{ijk}(t_j(n)_-), \tag{7.10}$$

Recall that $\tau_{ij}(n)$ is the age of the iteration value from iteration i used to update the n-th step of iteration j. In other words, for $\tau_{ij}(n)$, n is always the iteration counter associated with the iteration of the second lower index j. In (7.10), the worker AoI sequences are

$$\tilde{\tau}_{ijk}(t) \coloneqq \sum_{l=1}^{K_i} \left(N_{il}(t) - N_{il}(t - B_{jk}(t))\right). \tag{7.11}$$

Further, $t_j(n) \coloneqq \max_{n_1+\ldots+n_{K_j}=n} S_{jk}(n_k)$ is the point in continuous time, where the n-th update of coordinate j occurs.

Algorithm 3 Generic coordinate-wise Asynchronous Parameter Server Iteration (cAPSI)

```
Initialize the parameter iterations.
for the entire duration do
    for each worker do
        Receive parameter update job.
        Complete other jobs in the queue.
        Compute the parameter update.
        Send the update to the associated iteration.
    for each parameter iteration do
        Receive an update step from a worker.
        Update the parameter iteration.
        Collect current parameters from the other iterations.
        Sent a new job to the worker.
```

7.3 Main Results

The following two subsections present the main results for the APSI AoI sequence (7.9) and the cAPSI AoI sequences (7.10).

7.3.1 AoI weak limit analysis for APSI

We begin with the results for the APSI AoI sequence (7.9). The first lemma presents the asymptotic probability with which a worker performs an update. To state the lemma, define the events

$$E_k(n) \coloneqq \{n\text{-th update by worker } k\} = \{t(n) = S_k(N_k(t(n)))\} \tag{7.12}$$

Lemma 7.1. $\lim_{n\to\infty} \mathbb{P}\left(E_k(n)\right) = \frac{\frac{1}{\mu_k}}{\sum_j \frac{1}{\mu_j}} =: a_k.$

The first main theorem derives the limiting AoI distribution for each worker AoI sequence $\tilde{\tau}_k(t)$ conditioned on the event that worker k applies an update to the parameter server at time t. These time steps define the parameter server sequence AoI (7.9). Thus in combination with Lemma 7.1, we obtain the limiting distribution for $\tau(n)$ as the mixture distribution of the limiting conditional worker AoI sequences.

Theorem 7.2. *$\tau(n)$ converges weakly to a random variable τ with a mixture distribution of $\sum_{j\neq k}\tilde{N}_j(W_k)$ with weights a_k from Lemma 7.1, where*

$$\tilde{N}_j(h) \sim \lim_{t\to\infty}\left(N_j(t) - N_j(t-h)\right) \tag{7.13}$$

is the stationary renewal process associated with $N_j(t)$ (Definition 0.0.11).

Proof. The first step is to show that conditioned on the event sequence

$$\{t(n) = S_k(N_k(t(n)))\},$$

the worker AoI sequence $\tilde{\tau}_k(t(n))$ converges weakly to $\sum_{j\neq k}\tilde{N}_j(W_k)$. This step uses the weak convergence from Blackwell's renewal theorem (see A.2) and the independence of the worker renewal processes. We provide the details in the chapter appendix.

To complete the proof, we combine the conditional week convergence with Lemma 7.1. Let f be a bounded, continuous real-valued function, then

$$\begin{aligned}\mathbb{E}\left[f(\tau(n))\right] &= \sum_{k=1}^{K}\mathbb{E}\left[f\left(\tau_k(t(n))\right) \mid t(n) = S_k(N_k(t(n)))\right]\mathbb{P}\left(t(n) = S_k(N_k(t(n)))\right)\\ &\to \sum_{k=1}^{K} a_k\mathbb{E}\left[f\left(\sum_{j\neq k}\tilde{N}_j(W_k)\right)\right] = \mathbb{E}\left[f(\tau)\right],\end{aligned} \tag{7.14}$$

for a random variable τ as stated in the Theorem. The first equality follows by the law of total expectation. The limit follows by the conditional weak convergence from the first part of the theorem, Portmanteaus Theorem (see A.2) and Lemma 7.1. Finally, the last equality follows from elementary properties of mixture densities; see, e.g., (Frühwirth-Schnatter 2006, Sec. 1.2.4).

□

Based on Theorem 7.2, we derive several moment properties. Naturally, we require sufficient uniform integrability to obtain moment convergence from weak convergence.

Lemma 7.3. *Suppose that $\mathbb{E}\left[W_k^{q_k}\right] < \infty$ for $q_k > 1$. Then, $\tau(n)^q$ is uniformly integrable with $q \coloneqq \min_k q_k$.*

We now derive the limiting mean AoI. As announced, the limiting mean AoI is proportional to the number of workers and independent of the service time distributions. We utilize a well-known theorem for renewal processes with stationary increments, which also holds more generally for point processes with stationary increments.

Theorem 7.4 ((Serfozo 2009, Prop. 75))**.** *If N is stationary point process and $\mathbb{E}\left[N(1)\right] < \infty$, then $\mathbb{E}\left[N(t)\right] = t\mathbb{E}\left[N(1)\right], t \geq 0$.*

Corollary 7.5.

$$\lim_{n\to\infty} \mathbb{E}\left[\tau(n)\right] = \sum_{k=1}^{K} a_k \left(\sum_{j\neq k} \frac{\mu_k}{\mu_j} \right) = K - 1.$$

Proof. By Theorem 7.2, Lemma 7.3 and linearity of expectation, we have that

$$\lim_{n\to\infty} \mathbb{E}\left[\tau(n)\right] = \sum_{k=1}^{K} a_k \left(\sum_{j\neq k} \mathbb{E}\left[\tilde{N}_j(W_k)\right] \right). \tag{7.15}$$

Then apply Theorem 7.4, the stationarity of $\tilde{N}_j$ and its independence of W_k to conclude that

$$\lim_{n\to\infty} \mathbb{E}\left[\tau(n)\right] = \sum_{k=1}^{K} a_k \left(\sum_{j\neq k} \frac{\mu_k}{\mu_j} \right) \tag{7.16}$$

$$= \frac{1}{\sum_{k=1}^{D} \frac{1}{\mu_k}} \sum_{k=1}^{K} \left(\sum_{j\neq k} \frac{1}{\mu_j} \right) \tag{7.17}$$

$$= K - 1 \tag{7.18}$$

□

Next, we derive a general moment bound for which we first prove an auxiliary lemma that uses the subadditivity of each $\tilde{N}_j$ in p-norm.

Lemma 7.6. *Consider the p-norms*

$$\|\tilde{N}_j(W_k)\|_p := \mathbb{E}\left[\tilde{N}_j(W_k)^p\right]^{\frac{1}{p}} \tag{7.19}$$

then,

$$\|\tilde{N}_j(W_k)\|_p \leq \|\lceil W_k \rceil\|_p \|\tilde{N}_j(1)\|_p. \tag{7.20}$$

where $\lceil\cdot\rceil$ *denotes the ceiling function.*

Proposition 7.7. *Suppose that* $\mathbb{E}\left[W_k^{q_k}\right] < \infty$ *for* $q_k > 1$. *Then for all* $1 < p \leq \min_k q_k$, $\lim_{n\to\infty} \mathbb{E}\left[\tau(n)^p\right] = \mathbb{E}\left[\tau^p\right]$ *with*

$$\mathbb{E}\left[\tau^p\right] \leq \sum_{k=1}^{K} a_k \mathbb{E}\left[(W_k + 1)^p\right] \left(\sum_{j\neq k} \|\tilde{N}_j(1)\|_p \right)^p.$$

Proof. Let $1 < p \leq \min_k q_k$, then by the continuous mapping theorem, (Billingsley 2013, Thm. 2.7), $\tau(n)^p$ converges weakly to τ^p with τ from Theorem 7.2. As $\tau(n)^p$ is uniformly integrable, Lemma 7.3, (Billingsley 2013, Thm. 3.5) shows that $\lim_{n\to\infty} \mathbb{E}\left[\tau(n)^p\right] = \mathbb{E}\left[\tau^p\right]$. Therefore, as τ has mixture distribution,

$$\lim_{n\to\infty} \mathbb{E}\left[\tau(n)^p\right] = \sum_{k=1}^{K} a_k \mathbb{E}\left[\left(\sum_{j\neq k} \tilde{N}_j(W_k) \right)^p \right]. \tag{7.21}$$

The statement follows from (7.21), Lemma 7.6, and Minkowski's inequality. □

Remark 7.3.1. *For Proposition 7.7, note that workers may have a different largest bounded moment due to their heterogeneity. Recall that a bounded (non-central) moment implies the boundedness of all smaller (non-central) moments in probability spaces.*

Remark 7.3.2. *The terms $\|\tilde{N}_j(1)\|_p$ in Proposition 7.7 are the L^p norms of the number of renewals of the associated modified renewal process in the unit interval. For a given processing time distribution (or an empirically determined distribution), these terms may be readily computed numerically using*

$$\mathbb{P}\left(\tilde{N}_j(t) > m\right) = G_j * F_j^{m*}(t)\,, \tag{7.22}$$

with G as in Definition 0.0.11; see e.g. (Taga 1963). Further, one can also establish bounds for $\|\tilde{N}_j(1)\|_p$ using Chernoff bounds or the Azuma–Hoeffding inequality.

Since we used Minkowski's inequality and subadditivity, the moment bounds in Proposition 7.7 are loose. Complementary to Proposition 7.7, we derive a sharp upper bound for the second moment. The core part of this bound is first to establish an exact expression for the limiting second moment using conditional variances:

Definition 7.3.1. *For random variables X and Y, the conditional variance of X given Y is defined as $Var(X \mid Y) \coloneqq \mathbb{E}\left[(X - \mathbb{E}[X \mid Y])^2 \mid Y\right]$.*

Proposition 7.8. *Suppose that $\mathbb{E}\left[W_k^2\right] < \infty$ for all k, then*

$$\lim_{n\to\infty} \mathbb{E}\left[\tau(n)^2\right] = \sum_{k=1}^{K} a_k \left(\sum_{j\neq k} \mathbb{E}\left[Var\left(\tilde{N}_j(W_k) \mid W_k\right)\right] + \mathbb{E}\left[W_k^2\right] \left(\sum_{j\neq k} \frac{1}{\mu_j} \right)^2 \right)$$

Proposition 7.8 leads to exact expressions for the asymptotic second moment provided that the convolutions of the processing time distributions in (7.22) have tractable closed-forms. The simplest example arises for Poisson point processes.

Corollary 7.9. *Suppose the DC system is composed of workers with exponentially distributed processing times with $\mathbb{E}[W_k(n)] = \mu_k$, then for all $n \geq 0$,*

$$\begin{aligned}
\mathbb{E}\left[\tau(n)^2\right] &= \sum_{k=1}^{K} a_k \left(\sum_{j\neq k} \frac{\mu_k}{\mu_j} \right) \left(1 + 2 \left(\sum_{j\neq k} \frac{\mu_k}{\mu_j} \right) \right) \\
&= K - 1 + 2 \sum_{k=1}^{K} a_k \left(\sum_{j\neq k} \frac{\mu_k}{\mu_j} \right)^2
\end{aligned}$$

If the processing times are also identically distributed, then for all $n \geq 0$,

$$\mathbb{E}\left[\tau(n)^2\right] = (K-1)\left(1 + 2(K-1)\right)$$

Remark 7.3.3. *Proposition 7.8 exemplifies that, different from the first moment, higher-order moments of AoI due to APSI depend on the actual processing time distribution. Further, for Poisson processing times, Corollary 7.9 shows that the second moment is quadratic in the number of workers.*

For applications, only empirically approximated processing time distributions will be available, and common processing time distributions like the Pareto distribution do not have tractable closed-form convolutions. The announced sharp general bound for the second moment circumvents these issues. The bound follows from Proposition 7.8 and an inequality for the variance of stationary renewal processes.

Corollary 7.10.

$$\lim_{n\to\infty} \mathbb{E}\left[\tau(n)^2\right] \leq \sum_{k=1}^{K} a_k \left(\sum_{j\neq k} \frac{1}{4} \left(\frac{\mathbb{E}\left[W_j^2\right]}{\mu_j^2} \right)^2 + \frac{Var(W_j)\mu_k}{\mu_j^3} + \mathbb{E}\left[W_k^2\right] \left(\sum_{j\neq k} \frac{1}{\mu_j} \right)^2 \right)$$

Proof. For a stationary renewal process $N(t)$ with interarrival distribution W, Daryl J Daley (1978, Eq. 1.15) showed that

$$\mathrm{Var}\left(N(t)\right) \leq \frac{\mathrm{Var}\left(W\right)}{\mathbb{E}\left[W\right]^3} t + \frac{1}{4} \left(\frac{\mathbb{E}\left[W^2\right]}{\mathbb{E}\left[W\right]^2} \right)^2 . \tag{7.23}$$

The statement follows from Proposition 7.8 and the linearity in (7.23). □

Corollary 7.10 is sharp as Proposition 7.8 is errorless, and Daley's inequality holds with equality for deterministic renewal processes. Further, we will verify in Section 7.4 that Corollary 7.10 generally offers a small relative error for an arbitrary number of workers. Most notably, the relative error decreases as the number of workers K increases. This can be seen from Corollary 7.10; the third term is quadratic in the number of workers and will thus dominate the other two terms that are only linear in the number of workers.

Remark 7.3.4. *The asymptotic results in Theorem 7.2, Corollaries 7.5 and 7.10 and Proposition 7.7 hold for all $n \geq 0$ when modified renewal processes (Definition 0.0.11) are considered instead of ordinary renewal processes. In other words, when a DC system in equilibrium mode is observed that has already run for a long time.*

Remark 7.3.5. *We conjecture that a generalization of Proposition 7.8 to all higher-order integer moments is possible using higher-order cumulants and their relation to non-central moments. Please observe that the essence of Proposition 7.8 is to combine that the limit of $\tau(n)$ has mixture density and that the variance of conditionally independent random variables is the sum of their conditional variances. Thus, the key point will be to apply a generalization of the law of total variance for cumulants (Brillinger 1969).*

7.3.2 AoI weak limit analysis for cAPSI

The analysis of Algorithm 3 works along the lines of the previous subsection with some additional bookkeeping due to multiple parameter iterations. Recall that the key difference in the cAPSI AoI sequences $\tau_{ij}(n)$ (7.10) compared to the APSI AoI sequence (7.9) is that the number of renewals, while one worker computes an update, are from the workers on the other coordinates if $i \neq j$. As introduced in Section 7.2.2, a lower index jk denotes the k-th worker on the j-th coordinate. Analogously to Lemma 7.1 it now follows that

$$\lim_{n\to\infty} \mathbb{P}\left(E_{jk}(n)\right) = \frac{\frac{1}{\mu_{jk}}}{\sum_{l=1}^{K_j} \frac{1}{\mu_{jl}}} =: a_{jk}. \tag{7.24}$$

with

$$E_{jk}(n) \coloneqq \{n\text{-th update of coordinate } j \text{ by the } k\text{-th worker on coordinate } j\} \tag{7.25}$$

Theorem 7.11. *$\tau_{ij}(n)$ converges weakly to a random variable τ_{ij} with a mixture distribution of*

- *$\sum_{l\neq k} \tilde{N}_{il}(W_{jk})$ with weights a_{jk} if $i = j$,*
- *$\sum_{l=1}^{K_i} \tilde{N}_{il}(W_{jk})$ with weights a_{jk} if $i \neq j$,*

with the stationary renewal process associated with $N_{il}(t)$:

$$\tilde{N}_{il}(h) \sim \lim_{t\to\infty} \left(N_{il}(t) - N_{il}(t-h)\right).$$

Proof. Analogous to Theorem 7.2. □

Corollary 7.12.

$$\lim_{n\to\infty} \mathbb{E}\left[\tau_{ij}(n)\right] = \frac{\sum_{k=1}^{K_i} \frac{1}{\mu_{ik}}}{\sum_{k=1}^{K_j} \frac{1}{\mu_{jk}}} \times \begin{cases} K_j & i \neq j, \\ K_j - 1 & i = j, \end{cases} \tag{7.26}$$

Proof. The case $i = j$ follows by Corollary 7.5. Similarly for $i \neq j$, we have by Theorem 7.11 and Theorem 7.4 that

$$\lim_{n\to\infty} \mathbb{E}\left[\tau_{ij}(n)\right] = \sum_{k=1}^{K_j} a_{jk} \left(\sum_{l=1}^{K_i} \frac{\mu_{jk}}{\mu_{il}}\right) \tag{7.27}$$

$$= \frac{1}{\sum_{k=1}^{K_j} \frac{1}{\mu_{jk}}} \sum_{k=1}^{K_j} \left(\sum_{l=1}^{K_i} \frac{1}{\mu_{il}}\right) \tag{7.28}$$

□

Analogous statements to Proposition 7.7, Proposition 7.8, and Corollary 7.9 can now also be derived based on Theorem 7.11.

7.4 Numerical Verification

We now verify the core theoretical statements presented in this chapter by numerical simulations.

We begin verifying Corollaries 7.5 and 7.12. We simulate Algorithm 3 with three coordinates with $K_1 = 50$, $K_2 = 35$ and $K_3 = 20$ workers. The processing times of workers in each class are sampled from Pareto distributions with Pareto exponent $\alpha_1 = 2.1$, $\alpha_2 = 2.8$ and $\alpha_3 = 3.5$. Figures 7.1 to 7.3 show the resulting cumulative averages of the AoI sequences $\tau_{ij}(n)$. The cumulative averages converge to the asymptotic means of the AoI sequences by the SLLN for fundamental AoI processes Corollary 6.9[1]. In addition, the figures display the predicted asymptotic mean from Corollary 7.12. We see that the cumulative averages converge to their predicted values, which verifies Corollaries 7.5 and 7.12. Notice that in Figures 7.1 to 7.3, n is used as the index/counter for updating the corresponding coordinate. During the same runtime, coordinate one was updated most frequently, while coordinate three was updated least frequently. On the flip side, coordinate three has the lowest average AoI, while coordinate one has the largest average AoI. This highlights the challenge and trade-off in scheduling heterogeneous workers to update different coordinates of a model asynchronously. With Corollaries 7.5 and 7.12, the average AoI affecting jobs with asynchronous execution on parallel computing infrastructure can now be predicted exactly. The only required information is the mean processing time for a sequential job added to a specific worker, which will be readily available from infrastructure data (Samsi et al. 2021).

Second, we evaluate the quality of Corollary 7.10. Here, we consider APSI with homogenous workers with exponentially distributed processing times. In other words, parameter updates are completed at instances of Poisson point processes. Corollary 7.9 thus yields the exact second moment, and we can calculate the exact relative error between the bound in Corollary 7.10 and the actual value. We do this for varying Poisson rates λ and number of workers K, which leads to Figure 7.4. We can see that the relative error increases as λ decreases, in other words, when the workers get "slower". On the other hand, the relative error decreases as we increase the number of workers. This is a very attractive property, which we also expect for the anticipated generalization of Corollary 7.10 from higher-order cumulants.

[1]The SLLN extends from the fundamental AoI processes to AoI processes generated from point process. Here, the required mixing rate follows from Berbee (1987, Theorem 6.1) and a suitable approximation of the Pareto processing times by a Zeta distribution; the discrete analog

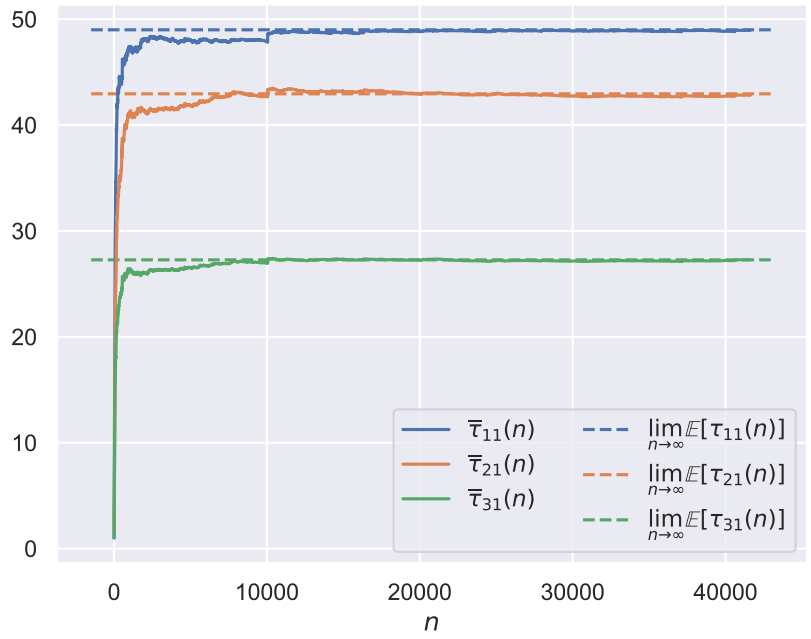

Figure 7.1: Cumulative average of the AoI sequences $\tau_{i1}(n)$, $1 \leq i \leq 3$ and the associated asymptotic mean AoI as predicted by Corollary 7.12.

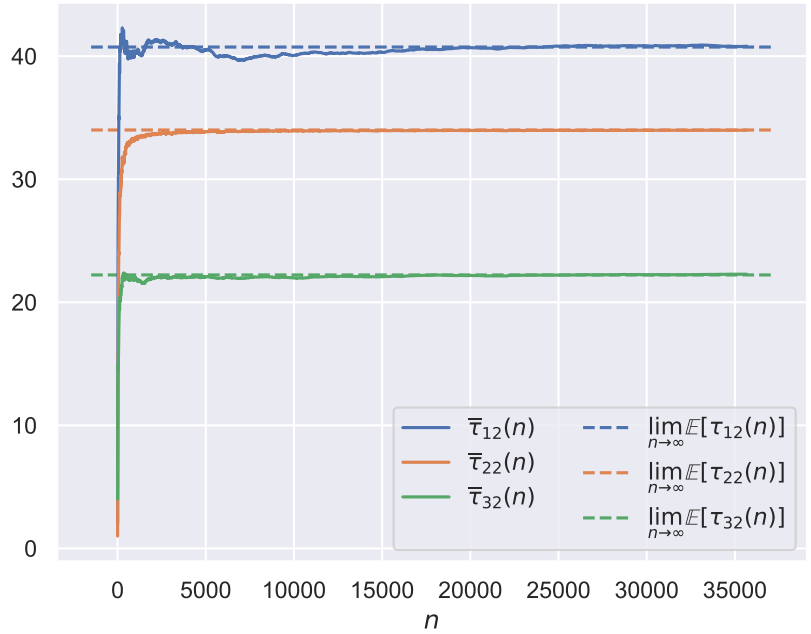

Figure 7.2: Cumulative average of the AoI sequences $\tau_{i2}(n)$, $1 \leq i \leq 3$ and the associated asymptotic mean AoI as predicted by Corollary 7.12.

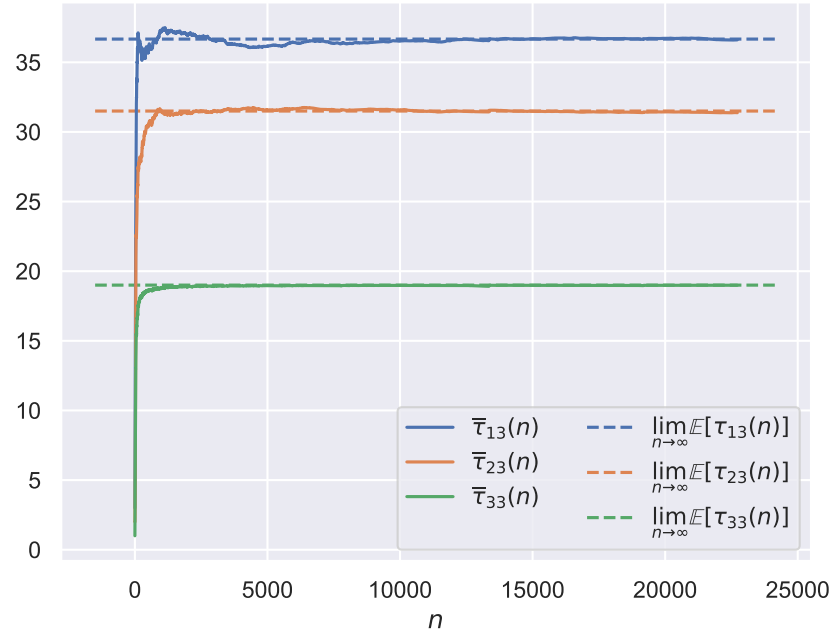

Figure 7.3: Cumulative average of the AoI sequences $\tau_{i3}(n)$, $1 \leq i \leq 3$ and the associated asymptotic mean AoI as predicted by Corollary 7.12.

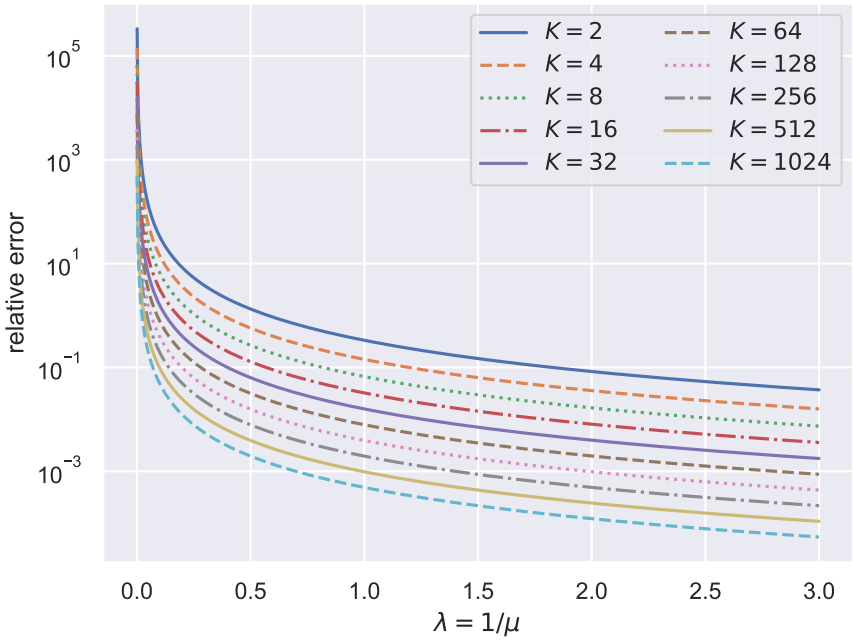

Figure 7.4: Relative error of the second moment bound in Corollary 7.10 for homogeneous workers with exponentially distributed processing times with rate λ

7.5 Applications

This section discusses conclusions from the results presented in the previous section. The main applications are to predict the convergence rate of methods and allocate computing resources. With this, we complete the answers on (Q8) on connections between computing events in continuous time and the effect on discrete algorithms.

7.5.1 Gradient descend methods

The prime examples of APSI are variants of ASGD. Until lately, finite time analyses of ASGD algorithms relied on the assumption of bounded AoI; see, e.g., the references in (Mishchenko et al. 2022, Sec. 1.1). This assumption was drastically weakened with delay-adaptive ASGD methods. These methods guarantee finite time convergence error estimates as a function of the first and/or second AoI moment; see (Cohen et al. 2021; Anastasiia Koloskova, Stich, and Jaggi 2022; Aviv et al. 2021).

The first inside for a distributed computing scenario, as considered herein, is that the convergence rates in (Mishchenko et al. 2022, Thm 3.3) and (Anastasiia Koloskova, Stich, and Jaggi 2022, Cor. 9) are asymptotically identical up to a constant factor. The first result gives a convergence rate in $\mathcal{O}\left(\frac{K}{n}+\frac{\sigma}{\sqrt{n}}\right)$, where σ^2 is a bound on the expected squared stochastic gradient error. The second result gives a convergence rate in $\mathcal{O}\left(\frac{\overline{\tau}_n}{n}+\frac{\sigma}{\sqrt{n}}\right)$ with $\overline{\tau}_n = \frac{1}{n}\sum_{t=1}^{n}\tau(n)$. It was mentioned in (Anastasiia Koloskova, Stich, and Jaggi 2022) that it was previously unclear how to compare these two results. By Corollary 7.5 and the obtained SLLN for AoI sequences, the rates are identical to a constant factor when workers complete ASGD updates after processing times modeled as parallel renewal processes.

In (Aviv et al. 2021), a powerful convergence rate estimate was proposed for a class of strongly convex objectives on bounded domains. The authors in (Anastasiia Koloskova, Stich, and Jaggi 2022) are, however, unclear about the result's strength as the estimate is a function of the delay second moment, which "can frequently degrade with the maximum delay". We shed new light on this. (Aviv et al. 2021, Thm. 4.3.) gives a converges rate estimate for the n-th step with a multiplicative factor $\sqrt{\sigma_n^2+\overline{\tau}_n{}^2}$, where σ_n^2 is the average AoI variance until step n. If the workers' processing times have finite second moments, then by Lemma 7.3 and the SLL, this quantity will converge to $\lim_{n\to\infty}\sqrt{\mathbb{E}\left[\tau(n)^2\right]}$. It thus follows from the second moment bound characterization, Proposition 7.8 and Corollary 7.10, that $\sqrt{\sigma_n^2+\overline{\tau}_n{}^2}$ is asymptotically equal to K up to constant factor plus an additional term in the order of $\sqrt{K}$. For example, if all workers have identically distributed processing time distributions, then

$$\sqrt{\sigma_n^2+\overline{\tau}_n{}^2} \in \Theta\left(\frac{\sqrt{\mathbb{E}\left[W^2\right]}}{\mathbb{E}\left[W\right]}(K-1)\right); \tag{7.29}$$

a similar factor can be obtained for heterogeneous workers. For given processing time data, the convergence rate for the delay-adaptive method in (Aviv et al. 2021) can therefore be assessed precisely. However, for workers with large processing time variance, the rate can be weak and will be inconclusive for fat-tailed processing time distributions with unbounded variance.

7.5.2 Resource allocation for parallel SGD iterations

We now use the AoI theory to design a DC system resource allocation problem based on the convergence result from (Aviv et al. 2021) for constraint convex optimization.

Suppose a DC system workload manager receives jobs in the form of datasets $\mathcal{D}_i$, pre-trained continuous feature extractors $\phi_i(x) \in \mathbb{R}^{d_i}$ and compact, convex constraint sets Θ_i for $i \in \{1, \dots, D\}$ and some $D \in \mathbb{N}$. The task is to solve the D constraint quadratic optimization problems:

$$\begin{aligned} \min_{\theta} \quad & \underbrace{\mathbb{E}_{(x,y)\sim\mathcal{D}_i}\left[\|\phi_i(x)^T\theta_i - y\|_2^2\right]}_{:=F_i(\theta_i)}, \\ \text{s.t.} \quad & \theta_i \in \Theta_i \subset \mathbb{R}^{d_i}, \end{aligned} \tag{7.30}$$

where for simplicity we assume $y \in \mathbb{R}$. In other words, the task is to find constraint linear regression models. Such tasks frequently arise in cloud computing scenarios where users submit a large dataset and a pre-trained model (e.g., neural network image feature extractor (Hinterstoisser et al. 2018)) and are looking for the best linear fit for their data. Suppose further that the DC system operator manages K heterogeneous workers. The goal is to schedule D asynchronous SGD algorithms as sequential tasks to the workers, such that all iterations achieve $\mathcal{O}(\varepsilon)$ accuracy in the least possible time. In other words, we want to minimize the average expected $\mathcal{O}(\varepsilon)$-makespan. For illustration, consider no online rescheduling and that $D \leq K$.

Assumption 7.5.1. *Each* $A_i := \mathbb{E}\left[\phi_i(x)\phi_i(x)^T\right]$ *is positive semi-definite matrix with matrix norm* $L_i := \|A_i\| < \infty$.

Define, $G_i := 2\max_{\theta_i\in\Theta_i}\|A_i\theta_i - \mathbb{E}\left[y\phi_i(x)\right]\|$ and $H_i = 2\lambda_{\min}(A_i)$, where $\lambda_{\min}$ denotes the smallest. Then F_i is H_i-strongly convex and L_i-smooth with $\|\nabla_{\theta_i}F_i(\theta_i)\| \leq G_i$. Further, assume a variance bound for the sample gradient: $\mathbb{E}\left[\|\nabla_{\theta_i}F_i(\theta_i) - \nabla_{\theta_i}f_i(\theta_i, x, y)\|^2\right] \leq \sigma^2\ \nabla_{\theta_i}f_i(\theta_i, x, y)$. Note that a finite subset of the data sets can be used to approximate these constants and check Assumption 7.5.1 quickly.

The assumptions of (Aviv et al. 2021, Algorithm 4 and Thm. 4.3) are satisfied, and their asynchronous adaptive SGD algorithm returns a candidate solution $\theta_i'(n)$ after n steps with an estimate

$$\mathbb{E}\left[\theta_i'(n) - \theta_i^*\right] \in \mathcal{O}\left(\frac{G_i^2+\sigma_i^2}{H_i n^2} + \frac{\sigma_i^2}{H_i n} + \frac{L_i^2(G_i^2+\sigma_i^2)}{H_i^3 n^2}\left(\sigma_n^2 + \overline{\tau}_n{}^2\right)\right). \tag{7.31}$$

It is left to assess $\sigma_n^2 + \overline{\tau}_n{}^2$ based on the parallel renewal process DC system model.

DC system workload managers like SLURM provide databases that track and log the processing time of specific job types on specific workers (Samsi et al. 2021). It is, therefore, realistic that an advanced/intelligent DC system manager can predict the rate at which certain jobs can be processed on specific workers, given the currently experienced system workload and local scheduling policies, e.g., generalized processor sharing. These rates can then be used as exponential approximations for the processing times of SGD jobs. In other words, let $\frac{1}{\mu_{ik}}$ be the predicted rate for an update step of the i-th problem by worker k.

Based on Corollary 7.9, the estimate for the convergence rate in (7.31) is therefore in

$$\mathcal{O}\Bigg(\underbrace{\frac{G_i^2+\sigma_i^2}{H_i n^2} + \frac{\sigma_i^2}{H_i n} + \frac{L_i^2(G_i^2+\sigma_i^2)}{H_i^3 n^2}\mathbb{E}\left[\tau_i^2\right]}_{:=g_i(n,\mathcal{K}_i)}\Bigg), \tag{7.32}$$

with

$$\mathbb{E}\left[\tau_i^2\right] = (|\mathcal{K}_i|-1) + 2\frac{\sum_{k\in\mathcal{K}_i}\mu_{ik}\left(\sum_{j\in\mathcal{K}_i\setminus\{k\}}\frac{1}{\mu_{jk}}\right)^2}{\sum_{k\in\mathcal{K}_i}\frac{1}{\mu_{ik}}},$$

where $\mathcal{K}_i$ is the set of workers allocated to SGD iteration i.

Now let $\varepsilon > 0$, then given (7.32) there exist a smallest n_i, such that $g_i(n_i, \mathcal{K}_i) \leq \varepsilon$. Further, let $\mathcal{B}_D$ denote the set of all partitions of $\{1, \ldots, D\}$. A combinatorial optimization problem to schedule workers for the SGD iterations to solve all (7.30) and to minimize the average expected $O(\varepsilon)$-makespan based on the given problem data and worker rate predictions can now be stated as:

$$\begin{aligned} \min_{(\mathcal{K}_1,\ldots,\mathcal{K}_D)\in\mathcal{B}_D} \quad & \frac{1}{D}\sum_i \min_{n_i} \mathbb{E}\left[t_{\mathcal{K}_i}(n_i)\right], \\ \text{subject to} \quad & g_i(n_i,\mathcal{K}_i) \leq \varepsilon, \end{aligned} \tag{7.33}$$

with the makespans $t_{\mathcal{K}_i}(n_i)$ for workers in $\mathcal{K}_i$ as defined (7.7).

Problem (7.33) presents a general structure for DC resource allocation problems: A runtime-related criterion (e.g., average makespan, maximum makespan, makespan variance, etc.) should be minimized subject to the fact that the executed algorithms satisfy a particular quality criterion (e.g., the expected distance to the optimal point), which depends on the AoI induced by allocated asynchronous resources. In this way, various problems like (7.33) can be formulated for non-convex and federated learning scenarios, e.g., based on (Anastasiia Koloskova, Stich, and Jaggi 2022, Thm. 11).

7.6 Discussion and related work

In this chapter, we derived the limiting distribution and moment bounds for AoI resulting from asynchronous parameter server updates modeled as parallel renewal processes. Based on the moment predictions, one can assess the performance of asynchronous algorithms and design resource allocation problems for asynchronous algorithms running on DC systems, as illustrated in the previous section. In the future, we intend to find practical approximations for such resource allocation problems for general non-convex optimization algorithms. One challenge that arises is the makespan expression itself, which is the maximum over random variables. For example, in problem (7.33), the makespan is the maximum over independent Erlang random variables due to the Poisson rate approximation. In this case, the expected makespan can be approximated by numerical integration, and extreme value theory will be helpful to approximate the maximum (Haan and Ferreira 2006; Gasull, López-Salcedo, and Utzet 2015).

As presented herein, DC systems modeled as parallel renewal processes have previously not been considered as a model for asynchronous parallel computing. In that sense, the established results are fundamental and new to the literature. In the queuing theory literature, multi-server queuing and computing have a long history since the seminal work of Erlang on M/D/K queues; see Kingman (2009) fresh look at Erlang's work. Since then, M/G/K queues with FCFC scheduling have been used as representative models for DC systems (Khazaei, J. Misic, and V. B. Misic 2011). The literature on M/G/K queues with FCFC scheduling offers good approximations for the mean waiting time and thus for the mean sojourn time (Gupta and Osogami 2011). However, the information delay that we call AoI is a metric that follows from the asynchronous computing of processors in a DC system. The AoI is thus a non-trivial function of the sojourn time, the job scheduler, and various other factors. The foundational contribution is to study AoI as a function of the processing times associated with jobs sequentially added to a DC system. This allows for a precise study of the AoI distribution. With this result, models for DC system queues, (Khazaei, J. Misic, and V. B. Misic 2011; Bandi, Trichakis, and Vayanos 2019), can be used to predict processing time properties for sequential tasks. The processing time properties can then be coupled with the results herein to characterize AoI properties, which in turn can then be used as design criteria for resource scheduling in DC systems.

Finally, recall that we derived the results of this chapter for the parallel renewal process, described by i.i.d. sequences of processing times. This was useful for deriving statistical properties for AoI from the core results of this section. Note, that the core theorems, Theorem 7.2 and Theorem 7.11, can generally hold for parallel point process. For Theorem 7.2, the only condition for this conclusion is that conditioned on the event sequence $\{t(n) = S_k(N_k(t(n)))\}$, the worker AoI sequence $\tilde{\tau}_k(t(n))$ converges weakly to some limit. This will hold for parallel point processes with asymptotically stationary (Cox and Isham 1980).

7.7 Proofs for Chapter 7

Proof of Lemma 7.1. We present a proof for $K = 2$ workers. For $E_k(n)$, we have that

$$\mathbb{P}\left(E_1(n)\right) = \mathbb{P}\left(A_2(t(n-1)) - A_1(t(n-1)) > 0\right), \tag{7.34}$$

which is the probability that the forward recurrence time (Definition 0.0.12) of worker 1 at the $n-1$-th update is less than the forward recurrence time of worker 2.

Next, write the distribution of $A_2(t(n-1)) - A_1(t(n-1))$ using the convolution formula for the difference of independent random variables.

$$\begin{aligned}
\mathbb{P}\left(E_1(n) \mid E_1(n-1)\right) &= \int_0^\infty \int_0^y f_{W_1}(x) f_{A_2(t(n-1))}(y) dxdy && (7.35)\\
&= \int_0^\infty \mathbb{P}\left(W_1 \leq y\right) f_{A_2(t(n-1))}(y) dxdy && (7.36)\\
&\to \frac{1}{\mu_2} \int_0^\infty \mathbb{P}\left(W_1 \leq y\right) \mathbb{P}\left(W_2 > y\right) dxdy && (7.37)\\
&= 1 - \frac{1}{\mu_2} \mathbb{E}\left[\min(W_1, W_2)\right], && (7.38)
\end{aligned}$$

with $\mu_{min_{12}} \coloneqq \mathbb{E}\left[\min(W_1, W_2)\right]$. In the same way, we can show that

$$\lim_{n\to\infty} \mathbb{P}\left(E_1(n) \mid E_2(n-1)\right) = \frac{\mu_{min_{12}}}{\mu_1}. \tag{7.39}$$

Using the law of total probability, it follows that

$$\lim_{n\to\infty} \mathbb{P}\left(E_1(n)\right) = \frac{\mu_{min_{12}}}{\mu_1} \lim_{n\to\infty} \mathbb{P}\left(E_2(n)\right) + \left(1 - \frac{\mu_{min_{12}}}{\mu_2}\right) \lim_{n\to\infty} \mathbb{P}\left(E_1(n)\right). \tag{7.40}$$

Equivalently,

$$0 = \frac{1}{\mu_1} \lim_{n\to\infty} \mathbb{P}\left(E_2(n)\right) - \frac{1}{\mu_2} \lim_{n\to\infty} \mathbb{P}\left(E_1(n)\right), \tag{7.41}$$

as $\mu_{min_{12}} > 0$. The statement now follows by solving the system of equations (7.41) and

$$\lim_{n\to\infty} \mathbb{P}\left(E_1(n)\right) + \lim_{n\to\infty} \mathbb{P}\left(E_2(n)\right) = 1. \tag{7.42}$$

The cases $K > 2$ follow the same principles, leading to K equations, respectively. □

Proof. details for Theorem 7.2. Let f be a bounded continuous function. Recall the definition of the worker AoI in continuous time (7.5) and observe that the event $\{t(n) = S_k(N_k(t(n))\}$ (that the n-th update is made by worker k), implies that the backward recurrence time $B_k(t(n)) = W_k(N_k(t(n)))$. It follows that

$$\mathbb{E}\left[f\left(\tilde{\tau}_k(t(n)_-)\right) \mid t(n) = S_k(N_k(t(n))\right] = \mathbb{E}\left[f\left(\sum_{j\neq k} K_{j,k,n}\right) \mid t(n) = S_k(N_k(t(n))\right] \tag{7.43}$$

with

$$K_{j,k,n} \coloneqq N_j(t(n)) - N_j\Big(t(n) - W_k(N_k(t(n)))\Big) \tag{7.44}$$

Now write the aforementioned event as the following disjoint union

$$\{t(n) = S_k(N_k(t(n))\} = \bigsqcup_{m\geq 0} \{t(n) = S_k(m)\}, \tag{7.45}$$

For each event on the right-hand side of (7.45), we now have for every $n \geq 0$ that

$$\begin{aligned}\mathbb{E}\left[f\left(\sum_{j\neq k} K_{j,k,n}\right) \mid t(n) = S_k(m)\right] &= \mathbb{E}\left[f\left(\sum_{j\neq k} N_j(S_k(m)) - N_j(S_k(m-1))\right)\right] \\ &\to \mathbb{E}\left[f\left(\sum_{j\neq k} \tilde{N}_j(W_k)\right)\right], \text{ as } m \to \infty.\end{aligned} \tag{7.46}$$

The convergence follows from Blackwell's renewal theorem using that the renewal processes are independent, the law of total expectation, and a truncation of the resulting summation as f is bounded.

Finally, we conclude from (7.43) and (7.45) that

$$\mathbb{E}\left[f\left(\tilde{\tau}_k(t(n)_-)\right) \mid t(n) = S_k(N_k(t(n))\right] = \sum_{m\geq 0} a_{n,m}\mathbb{E}\left[f\left(\sum_{j\neq k} K_{j,k,n}\right) \mid t(n) = S_k(m)\right], \tag{7.47}$$

with coefficients

$$a_{n,m} \coloneqq \frac{\mathbb{P}\left(t(n) = S_k(m)\right)}{\mathbb{P}\left(t(n) = S_k(N_k(t(n)))\right)} > 0$$

and $\sum_m a_{n,m} = 1$. Further, as $t(n) \to \infty$ a.s., as $S_k(m) < \infty$ a.s., and since the denominator of the coefficients converges by Lemma 7.1, we see that $a_{n,m} \to 0$ for every $m \in \mathbb{N}$ as $n \to \infty$. The theorem statement, consequently, follows from (7.46) and (7.47). □

Proof of Lemma 7.3. By Theorem 7.2, the independence of the parallel renewal processes and the definition of $\tau(n)$, it is enough to show that $(N_j(t) - N_j(t - W_k))^q$ is uniformly integrable for $\mathbb{E}\left[W_k^q\right] < \infty$. To see this, we can use the distributional subadditivity of each $N_j(t) + 1$, see (Iksanov 2016, Eq. (3)), which shoes that

$$\mathbb{P}\left(N_j(t) - (N_j(t-h) > x\right) \leq \mathbb{P}\left(N_j(h) + 1 > x\right). \tag{7.48}$$

Hence,

$$\mathbb{P}\left(N_j(t) - (N_j(t-W_j) > x\right) \leq \sum_h \mathbb{P}\left(N_j(h) + 1 > x\right)\mathbb{P}\left(W_k = h\right) \tag{7.49}$$

and it follows that $N_j(t) - (N_j(t - W_k)$ is uniformly stochastically dominated by random variable with bounded q-th moment as $\mathbb{E}\left[W_k^q\right] < \infty$. Using this uniform stochastic domination property, the required uniform integrability follows. □

Proof of Lemma 7.6. Inspired by the proof of Theorem 7.4, we show that $\|\tilde{N}_j(w)\|_p$ is a subadditive function:

$$\begin{aligned}\|\tilde{N}_j(u+v)\|_p &= \|\tilde{N}_j(u) + \tilde{N}_j(u+v) - \tilde{N}_j(u)\|_p \\ &\leq \|\tilde{N}_j(u)\|_p + \|\tilde{N}_j(u+v) - \tilde{N}_j(u)\|_p \\ &= \|\tilde{N}_j(u)\|_p + \|\tilde{N}_j(v)\|_p \end{aligned} \tag{7.50}$$

The inequality follows by Minkowski's inequality and the second equality by stationarity. Hence, $\|\tilde{N}_j(nw)\|_p \leq n\|\tilde{N}_j(w)\|_p$ for $n \in \mathbb{N}$. Elementary properties of subaddtive functions, see e.g. (Kuczma 2009), thus yield that $\|\tilde{N}_j(w)\|_p \leq n\|\tilde{N}_j(w/n)\|_p$. As, $\|\tilde{N}_j(w)\|_p$ is increasing it follows that

$$\|\tilde{N}_j(w)\|_p \leq \lceil w \rceil \|\tilde{N}_j(w/\lceil w \rceil)\|_p \leq \lceil w \rceil \|\tilde{N}_j(1)\|_p, \tag{7.51}$$

where $\lceil \cdot \rceil$ denotes the ceiling function. Using the law of total probability and the independence of $\tilde{N}_j$ and W_k it follows that

$$\|\tilde{N}_j(W_k)\|_p \leq \|W_k + 1\|_p \|\tilde{N}_j(1)\|_p. \tag{7.52}$$

□

Proof of Proposition 7.8. Starting from (7.21), using the well-known identity for the variance of random variables, we obtain

$$\lim_{n\to\infty} \mathbb{E}\left[\tau(n)^2\right] = \sum_{k=1}^{K} a_k \left(\operatorname{Var}\left(\sum_{j\neq k} \tilde{N}_j(W_k) \right) + \mathbb{E}\left[\sum_{j\neq k} \tilde{N}_j(W_k) \right]^2 \right) \tag{7.53}$$

Then, the law of total variance (see A.2) shows that

$$\operatorname{Var}\left(\sum_{j\neq k} \tilde{N}_j(W_k) \right) = \mathbb{E}\left[\operatorname{Var}\left(\sum_{j\neq k} \tilde{N}_j(W_k) \mid W_k \right) \right] + \operatorname{Var}\left(\mathbb{E}\left[\sum_{j\neq k} \tilde{N}_j(W_k) \mid W_k \right] \right). \tag{7.54}$$

As $\tilde{N}_j(W_k)$ are conditionally independent given W_k it follows that

$$\operatorname{Var}\left(\sum_{j\neq k} \tilde{N}_j(W_k) \right) = \sum_{j\neq k} \mathbb{E}\left[\operatorname{Var}\left(\tilde{N}_j(W_k) \mid W_k \right) \right] + \operatorname{Var}(W_k) \sum_{j\neq k} \left(\frac{1}{\mu_j} \right)^2, \tag{7.55}$$

For the second term, we again used the stationary of $\tilde{N}_j$, the independence of $\tilde{N}_j$ and W_k, as well as the identity $\operatorname{Var}(cX) = c^2 = \operatorname{Var}(X)$ for a random variable X. The proposition now follows by combining (7.53) and (7.55). □

Chapter 8

AoI in wireless networks

In this last chapter on AoI, we study conditions to verify the existence of AoI dominating random variables in (mobile) wireless networks. Based on these conditions, stability and convergence guarantees for the distributed SA and optimization algorithms, as studied in Chapters 2 to 4, become available when implemented over wireless networks. The core examples of such scenarios are mobile computing systems consisting of robots and drones equipped with sensor units. In these scenarios, iterative decentralized algorithms are attractive as they facilitate online learning adapted to the uncertainty in a sensor data stream. Further, information exchanged via a wireless network facilitates cooperation among physically distributed agents. Wireless networks are popular because they are easy to set up, cheap, and flexible. However, they are not reliable. As discussed before, uncertain communication may lead to large AoI, which in turn may affect algorithm performance. To predict algorithm performance, it is thus important to assess AoI based on representative wireless network communication. A representative wireless network should include the correlation between communications that are close to each other in some domain (e.g., time, frequency, or space) and the use of Medium Access Control (MAC) protocols (Tse and Viswanath 2005).

This chapter discusses a novel *network model based on the signal-to-interference-plus-noise ratio (SINR)* associated with agent-to-agent communication that encompasses many wireless network configurations. Moreover, we present a novel *MAC protocol that accounts for "local" AoI-aware MAC decisions*. By local, we mean that the decisions are made at the agent level, depending on observable AoI. The SINR model encompasses deterministic path-loss models, uncertain components like shadowing, fading, interference by other agents, and additional noise and interference from external users. These components may be time-varying and influence each other. We allow these components to vary with an underlying local network-state process. Such processes can, for example, be used to model agent mobility. Suppose we assume that the local network-state processes are geometrically ergodic Markov processes. In that case, in conjunction with the MAC

protocol, the network guarantees that the SINR for agent-to-agent communication is such that the various dependencies decay over time. Thus adding to the answers on (Q7) presented in Chapter 6. This result was presented in (Redder, Ramaswamy, and Karl 2022e). With the derived SINR dependency decay, we then bound the asymptotic growth of AoI variables associated with the communication based on the results in Chapter 6 on AoI with dependent communication and the growth bounds developed in Chapter 5. The asymptotic growth of the AoI variables then implies the stability and convergence of distributed stochastic gradient descent schemes implemented over wireless networks using the results presented in Chapter 3, which provides another dimension to answer of (Q4).

To the author's knowledge, this is the first time that freshness of data is directly derived using a practical wireless communication model within the distributed optimization setting. Previous analyses assume unrealistic models, e.g., i.i.d. communication, (Anastasia Koloskova et al. 2020; Ramaswamy, Redder, and Daniel E. Quevedo 2021b). We extend the theory of distributed optimization to allow for practical wireless communication models. Through this analysis, we believe that we have created a useful framework for practitioners and theoreticians.

8.1 Network model

Denote a set of agents by $V \coloneqq \{1, \dots, D\}$. As before, we use "$i, j$" to index the agents, and we are interested in the communication necessary for a distributed gradient-based solution to an optimization or learning problem as in Chapters 2 to 4. Hence, consider a scenario wherein agents are geographically distributed and do not know the other agents' current optimization variables. In particular, at time n, agent i does not know the optimization variables x_n^j for $j \neq i$. Instead, the agents use a wireless network to exchange updates to their local variables. Therefore, as in (4.8) in Chapter 4, only delayed versions of the other agent variables are available, i.e., agent i has access to the delayed version

$$X_n^i \coloneqq \left(x^1_{n-\tau_{i1}(n)}, \dots, x^i_n, \dots, x^D_{n-\tau_{iD}(n)}\right) \tag{8.1}$$

of the true current global variable x_n at any time step n. Here, we refer to X_n^i as the *local belief vector* of agent i at time n. The local belief vector then gets updated whenever an agent accesses the wireless network and successfully transmits information about the optimization to other agents. How the agents access the network is described by a MAC protocol.

Generally speaking, MAC protocols are necessary for wireless communication systems to avoid unnecessary data losses via controlled medium access. We use a time-slotted broadcast communication strategy to exchange the belief vectors X_n^i (8.1). At every step, each agent i uses a MAC protocol to decide whether it broadcasts X_n^i in time slot n. This decision is represented by p^i, which equals 1 when the decision is to send and 0 otherwise. The MAC protocol to be presented

herein, in conjunction with a suitable network, should facilitate the flow of x_n^i (local variable) from agent i to every other agent. It does so by facilitating the flow of the belief vectors X_n^i containing the local variables x_n^i, which, with suitable assumptions, will ensure that eventually, all agents i receive updates of x_n^j from all agents j.

Suitable assumptions for a network model should consider that wireless transmissions by agents occurring close in some domain will affect each other. In other words, a representative network model for multi-agent wireless communication should take into account the interference due to the network usage of other agents. Physically, an agent can receive data from another agent using a wireless channel if the received SINR of the observed signal is above a certain threshold. Iyer, Rosenberg, and Karnik (2009) argued that the SINR is thus a realistic and representative model for the success of wireless communication in the presence of interference. Below, we present a SINR model that can be used to model various practical wireless communication phenomena.

8.1.1 SINR-based network model

We associate with each agent i a stationary (network) state process $\{\Psi_n^i\}_{n\geq 0}$. These processes represent the local network environment at each agent. For each agent i, we have

$$\Psi_n^i \in \mathcal{Y}^i, \quad \forall n \geq 0, \tag{8.2}$$

with state space $\mathcal{Y}^i$. For example, Ψ_n^i may represent the physical position of agent i in $\mathbb{R}^2$ at time n. The following is the channel gain associated with the wireless channel from node i to j during time slot n:

$$g_n^{ij} := G_n^{ij}(\Psi_n^i, \Psi_n^j), \tag{8.3}$$

where $G_n^{ij} : \mathcal{Y}^i \times \mathcal{Y}^j \to \mathbb{R}_{\geq 0}$ are i.i.d. sequences of Borel measurable functions (Borel functions), which include continuous and indicator functions of the state processes. Further, we define local noise processes

$$\nu_n^i := G_n^i(\Psi_n^i), \tag{8.4}$$

where again $G_n^i : \mathcal{Y}^i \to \mathbb{R}_{\geq 0}$ are i.i.d. sequences of Borel functions. We then use an additive model for the wireless channel from agent i to j during time slot n

$$\mathrm{SINR}_n^{ij} := \frac{p_n^i g_n^{ij}}{\sum_{k\neq i} p_n^k g_n^{kj} + \nu_n^j}, \tag{8.5}$$

with transmission schedule $p_n^i \in \{0, 1\}$. Transmission scheduling is via an AoI-aware MAC protocol, discussed in Section 8.1.3.

The SINR model, (8.5), generalizes the traditional SINR model (Tse and Viswanath 2005). It includes popular models such as the following:

Example 8.1.1. *Let* $\mathcal{Y}^i = \mathbb{R}^2$. *Define*

$$\overline{SINR}_n^{ij} := \frac{p_n^i h_n^{ij} \|\Psi_n^i - \Psi_n^j\|^{-\gamma}}{\sum_{k \neq i} p_n^k h_n^{kj} \|\Psi_n^k - \Psi_n^j\|^{-\gamma} + \nu}, \tag{8.6}$$

with path-loss coefficient γ, *where each* h_n^{ij} *is an i.i.d. fading process and* ν *is constant background noise.*

It also includes more general models with:

1) General Borel measurable deterministic path-loss.
2) Slow (shadow) fading and fast fading.
3) Additional external noise or interference.

Notably, all these components can depend on each other via the local state processes Ψ_n^i and any additional i.i.d. process. Below, we illustrate that (8.5) can model state-dependent Rayleigh fading.

Example 8.1.2. *Suppose we would like to model state-dependent Rayleigh fading* h_n^{ij} *for* $\overline{SINR}_n^{ij}$ *in Example 8.1.1. To do so, we can generate independent standard normally distributed samples* u_n, v_n *and transform them linearly using a measurable function* g *of the network states:*

$$\tilde{u}_n := u_n g(\Psi_n^i, \Psi_n^i), \quad \tilde{v}_n := v_n g(\Psi_n^i, \Psi_n^i) \tag{8.7}$$

Then, $h_n^{ij} := \sqrt{(\tilde{u}_n)^2 + (\tilde{v}_n)^2}$ *will result in state-dependent Rayleigh fading, where the scale parameter of the Rayleigh distribution density depends on the network states.*

8.1.2 Network Assumptions

We assume a constant SINR threshold β such that, whenever agent i decides to send some data ($p_n^i = 1$) and

$$\mathrm{SINR}_n^{ij} \geq \beta, \tag{8.8}$$

the data from node i is successfully received at agent j during time slot n. The threshold depends on each node pair's modulation, coding, and path characteristics. Extending the analysis to node-pair-specific thresholds is simple, so assuming only one SINR threshold does not lose generality. The SINR will most importantly depend on the agents' physical position, the transmitter's transmission power, and the transmission power of other currently transmitting agents. The physical position will determine the channel gain between the receiver and all transmitting agents, e.g., due to multi-path propagation, shadowing, etc. For a simpler presentation, we will not consider additional communication latencies, e.g., latency due to coding, etc. Taking those into account would be possible without too many technical difficulties.

We use A_n^{ij} to represent the event sequence corresponding to (8.8), i.e. we define

$$A_n^{ij} \coloneqq \{\mathrm{SINR}_n^{ij} \geq \beta\} \tag{8.9}$$

Communication errors are due to interference when $\mathrm{SINR}_n^{ij} < \beta$. The threshold β is derived from the Shannon bound via the available bandwidth B and the data rate R required to exchange x_n during a time slot, i.e., $\beta > 2^{R/B} - 1$ (plus some additional safety margins, if desired). The extension to node-pair-specific margins is, again, a mere technicality. Hence, if A_n^{ij} occurs, agent i can successfully transmit its *entire* local belief vector X_n^i (comprising the most recent information from all agents available at agent i, including their local time stamps) from (8.1) to agent j during time slot n. Theoretically, arbitrary bit rates will merely require a lot of bookkeeping and will not change the results qualitatively.

We need a connectivity assumption enabling that eventually, all agents i receive updates of x_n^j from all agents j. We assume the following with regard to the wireless network for the analysis:

Assumption 8.1.1 (A1). *There is a strongly connected directed graph (V, E), i.e., a graph with a path between each ordered pair of vertices of the graph, such that for all $(i, j) \in E$, we have*

$$\mathbb{P}\left(g_{n\cdot m}^{ij} > \beta \nu_{n\cdot m}^{j}\right) > 0, \qquad \text{for all } n \geq 0 \text{ for some } m \in \mathbb{N}. \tag{8.10}$$

Assumption 8.1.2 (A2). *The network state process $\Psi \coloneqq (\Psi_1, \ldots, \Psi_D)$ forms a geometrically ergodic Markov chain (Definition 8.1.1).*

Definition 8.1.1 ((S. P. Meyn and Tweedie 2012, Ch. 15))**.** *A stationary Markov chain with transition kernel P is said to be* Geometrically Ergodic *if P_n convergences in total variation at a geometric rate to a unique stationary distribution.*

8.1.3 AoI-aware medium access control protocols

Assumption 8.1.1 and Assumption 8.1.2 enable the design of broadcast schedules $\{p_n^i\}_{n\geq 0}$ to guarantee that belief vectors can flow between every pair of agents. A simple schedule that suffices is a centralized round-robin schedule. Since a single agent transmits in a given time slot, Assumption 8.1.1 would guarantee that the transmission is successful with non-zero probability over a finite horizon. The recurrence of the network state process due to Assumption 8.1.2 then guarantees that this event occurs infinitely often.

We are interested in a more sophisticated schedule that takes decisions locally at the agent level based on the observed AoI processes. In other words, we would like to design an *AoI-aware MAC protocol* (ideally fully distributed) to facilitate fast convergence of the optimization algorithm at hand – in this case, a DSGD iteration as in Chapter 3. While this seems obvious from a practical perspective, for a theoretical analysis, it adds complications. It turns out that it is particularly

Algorithm 4 MAC protocol at agent $i \in V$

Fix $M_1 \geq M_2 \in \mathbb{N}$ and a small $\delta \in (0,1)$.
for the entire duration **do**
  for M_1 steps **do**
    Sample p_n^i from the AoI-aware policy Π_n^i
  for one time step **do**
    With probability δ pick $p_n^i = 0$.
    Elif $\tilde{\tau}_{ji}(n) > M_1$ for any $j \in \mathcal{N}(i)$, let $p_n^i = 1$.
  for M_2 steps **do**
    Pick p_n^i independent or deterministic.

difficult to analyze transmission schedules that are arbitrary functions of the AoI processes $\tau_{ij}(n)$. This is because the scheduling decisions $\{p_n^i\}_{n\geq 0}$ will depend recursively on previous scheduling decisions, which makes it difficult to analyze the SINR-driven event processes (8.9).

To overcome this, we present a MAC protocol (Algorithm 4) that periodically switches between an AoI-aware and an AoI-unaware component. The AoI-aware component itself does not distinguish the AoI when its value is "large". In this sense, the AoI-aware component is only quasi-cognizant. This slight compromise helps with the analysis while remaining practical.

Let M_1 represent the number of time steps used by the AoI-aware schedule, and let M_2 represent the number of steps used by the AoI-unaware schedule. The MAC protocol framework is stated as Algorithm 4. The algorithm uses the AoI associated with direct communication for agent pairs $(i,j) \in E$ with E from Assumption 8.1.1. For this, we assume that a sender receives an acknowledgment after successful transmission, i.e., if $p_n^i = 1$, then agent i receives feedback if A_n^{ij} occurred for any $j \neq i$. For agent $i \in V$ define $\mathcal{N}(i)$ the set of neighbors in (V,E). Then for any $j \in \mathcal{N}(i)$, define $\tilde{\tau}_{ji}(n)$ as the AoI associated with direct communication from i to j.

Let us now look at the AoI-aware component of Algorithm 4. For each agent $i \in V$, define local AoI averages:

$$q^i(n) := \frac{1}{(D-1)} \sum_{j \neq i} \min\{\tau_{ij}(n), M_2\} - 1 \tag{8.11}$$

$$r^i(n) := \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \min\{\tilde{\tau}_{ji}(n), M_2\} - 1 \tag{8.12}$$

Using a softmax function, we now define local time-varying stochastic AoI-aware policies Π_n^i by defining its density:

$$\Pi_n^i(1) := \frac{1}{1 + e^{\frac{1}{r^i(n)} - \frac{1}{q^i(n)}}} \tag{8.13}$$

Observe that if $p_n^i \sim \Pi_n^i$, then $\mathbb{P}\left(p_n^i = 1 \mid q^i(n) = 0\right) = 1$, i.e., if agent i has the best possible

status updates from all other agents, it will broadcast its own belief vector. By construction, the policy has the following properties:

- Agent i has a higher chance of transmitting at time n if it previously had less success in transmitting its local believe vector (hence, $\tilde{\tau}_{ji}(n)$ would be "large").
- Agent i has a lower chance of transmitting if it's own AoI processes $\tau_{ij}(n)$ take larger values. Hence, agent i would be more likely to remain silent if it had not received updates from other agents, as this may have been caused by its interference at other agents.

The time-varying AoI-aware policies Π_n^i are the essential component for the analysis. The key property is the following: *For $m > M_2$ the AoI-aware decisions do not distinguish events of the form $\{\tau_{ij}(n) > m\}$ from the event $\{\tau_{ij}(n) > M_2\}$.* Specifically, the AoI-aware schedules are Borel functions of the indicator functions $\mathbb{1}_{\{\tau_{ij}(n)>m\}}$ for $m = 0, \ldots, M_2$. To see that Π_n^i satisfies this property, observe that $\min\{\tau_{ij}(n), M_2\}$ can be represented as:

$$\sum_{k=1}^{M_2-1} k\mathbb{1}_{\{\tau_{ij}(n)>k-1\}}\mathbb{1}_{\{\tau_{ij}(n)\leq k\}} + M_2\mathbb{1}_{\{\tau_{ij}(n)>M_2\}} \tag{8.14}$$

In Lemma 8.3, we will use that Π_n^i is a Borel function $\mathbb{1}_{\{\tau_{ij}(n)>m\}}$ for $m = 0, \ldots, M_2$ to show that the dependency of events A_n^{ij} and A_m^{ij} decays as $|n - m| \to \infty$.

Finally, we look at the *backup* scheduling step, which completes the description of Algorithm 4. For a pair of agents $(i, j) \in E$, if agent j has not received an update from agent i during the AoI-aware phase, then this step ensures that with some probability, agent i will broadcast its belief vector. The backup step is a technical component that simplifies the proof of Lemma 8.2 in the next section. For the presented AoI-aware policies Π_n^i, a version of Lemma 8.2 can already be shown without it. This leads to the main result of this chapter – Theorem 8.1.

8.2 AoI moment bounds under Markov dynamics and AoI aware scheduling

Theorem 8.1. *Suppose every agent $i \in V$ uses Algorithm 4 to select its broadcast decisions p_n^i. Suppose the network can be represented by the SINR model (8.5), such that Assumption 8.1.1 and Assumption 8.1.2 hold. Then for any $p >$ there exists a random variable $\overline{\tau}$, such that $\tau_{ij}(n) \leq_{st} \overline{\tau}$ for all n, with $\mathbb{E}[\overline{\tau}^p] < \infty$.*

To simplify the proof of Theorem 8.1, we consider without loss of generality Assumption 8.1.1 with $m = 1$ for all i, j. We need the following lemmas to prove Theorem 8.1. Define $M := M_1 + M_2 + 1$ with M_1 and M_2 as in Algorithm 4.

Lemma 8.2. *For every $(i,j) \in E$, there is for some $\varepsilon > 0$:*

$$\mathbb{P}\left(\bigcup_{k=n}^{n+M} A_{ij}^{k}\right) \geq \varepsilon, \quad \forall n \geq 0. \tag{8.15}$$

8.2.1 Strong mixing preservation property

We now step into the theory of dependency decay. For this, recall the background on α-mixing presented in Section 6.2. By Assumption 8.1.2, the network state process Ψ converges to its steady state distribution at a geometric rate. It follows from (Bradley 2005, Thm. 3.7) that Ψ is α-mixing with exponential decay. The next lemma shows that this mixing property is preserved under the MAC protocol Algorithm 4.

Lemma 8.3. *A_n^{ij} is α-mixing with exponential decay.*

8.2.2 Moment bound

We are now ready to prove Theorem 8.1, which is practically a corollary to Theorem 6.1 developed in Chapter 6.

Fix a pair of agents $(i,j) \in E$, with E from Assumption 8.1.1. By Lemma 8.2, Theorem 6.1 applies with A_n^{ij} in place of A_n and $\kappa = M$, where by Lemma 8.3, $\alpha(A_{ij}, n)$ decays exponentially. It thus follows from Lemma Theorem 6.1 and the exponential decay that for every $p > 0$, there is a dominating random variable for each pair of agents that can communicate directly described by the directed graph (V, E) from Assumption 8.1.1. In addition, it is simple to see that the existence of a dominating random variable with some moment bound is a transitive property of multi-hop networks. It thus follows from the strong connectivity of (V, E) and a simple induction argument that for every $p > 0$ there is a single dominating random variable $\overline{\tau}$ such that $\overline{\tau}_{i,j} \leq_{\text{st}} \overline{\tau}$ for every $(i,j) \in V^2$, and $\mathbb{E}[\overline{\tau}^p] < \infty$. Thus yielding the statement of Theorem 8.1. The core property for this strong result is the geometric ergodicity assumption, which yields exponentially fast mixing.

Discussion on Assumption 8.1.1 and Assumption 8.1.2

Since assumptions Assumption 8.1.1 and Assumption 8.1.2 are key to the analysis, we discuss their practicality here. Assumption 8.1.1 requires that the wireless network guarantees that at a given agent pair can communicate with non-zero probability, *when all other agents are silent.* This assumption is weak and easy to ensure since it only requires that the communication is successful in the absence of interference. It is harder for the network to ensure successful communication in the presence of interference.

Given the SINR model (8.5), Assumption Assumption 8.1.1 is *necessary* to show Theorem 8.1 If Assumption 8.1.1 does not hold, then there is a pair of agents $(i,j) \in V^2$ such that for every path $(i_1,\dots,i_K)$, $K \geq 2$, with $i_1 = i$ and $i_k = j$, there is pair (i_k, i_{k+1}), such that (8.10) does not hold, i.e. $\mathbb{P}\left(g_n^{i_k i_{k+1}} > \beta \nu_n^{i_{k+1}}\right) = 0$ for all $n \geq 0$. Consequently, agent j will never receive information from agent i. This can be deduced using the stationarity of the network state process Ψ.

While necessary, Assumption 8.1.1 alone is not sufficient to guarantee 8.1. A trivial example where Assumption 8.1.1 is true would be to flip a fair coin once, at time 0. If heads, then all agents successfully communicate, all the time, with probability one. If tails, then there is no communication at all. Thus $\mathbb{P}\left(\tau_{ij}(n) = n \ \forall n \geq 1\right) = \frac{1}{2}$. In this example, $\lim_{|n-m|\to\infty} |\mathbb{P}\left(A_n^{ij} \cap A_m^{ij}\right) - \mathbb{P}\left(A_n^{ij}\right)\mathbb{P}\left(A_m^{ij}\right)| \neq 0$. Hence, the corresponding Ψ is not α-mixing. This necessitates Assumption 8.1.2.

Assumption Assumption 8.1.2 requires that the network state process Ψ in (8.5) is a geometrically ergodic Markov chain. It follows from (Bradley 2005, Thm. 3.7) that Ψ is α-mixing. In conjunction with Algorithm 4, it was therefore possible to show that $\mathbb{1}_{A_n^{ij}}$ is indeed also α-mixing (Lemma 8.3). Geometrically ergodic Markov chains are an important process class for wireless communication. For example, finite-state Markov chains have been used to approximate wireless fading channels: (G. Bianchi 2000; Lin et al. 2015). Notably, irreducible, aperiodic finite state Markov chains converge to their stationary distribution at a geometric rate. *However, Assumption 8.1.2 also accommodates infinite state space models that can represent mobile agents.*

Example 8.2.1. *Suppose each local state process $\Psi_n^i \in \mathbb{R}^m$, $m \geq 1$, has linear dynamics $\Psi_{n+1}^i = A^i \Psi_n^i + w_n^i$, with i.i.d. Gaussian noise w_n^i for a stable matrix $A^i \in \mathbb{R}^{m\times m}$, then Assumption 8.1.2 holds.*

8.3 Discussion and related work

Within the literature of distributed optimization, the following two types of network models are popular. First, networks with guaranteed periodic communication, e.g., (Scutari and Ying Sun 2019) and second, networks where the success of agent-to-agent communication is independent across agent pairs and over time, e.g., (Anastasia Koloskova et al. 2020). Neither of these models includes several practical components of wireless communication networks, e.g., the correlation between communications that are close to each other in some domain (e.g., time, frequency, or space) or the use of Medium Access Control (MAC) protocols.

The essence of the geometrically ergodic Markov chain assumption, Assumption 8.1.2, is to represent such correlations in a network process with a sufficient dependency decay, described by α-mixing. One can, therefore, simply replace the geometrically ergodic Markov chain model with an arbitrary α-mixing process. It is easy to see that the typical assumptions from the distributed

optimization literature (guaranteed periodic communication and communication described by independent events) directly imply α-mixing.

This chapter has discussed that a geometrically ergodic Markov chain can guarantee the required mixing properties to apply the results of Chapter 6. As an alternative to the Markov model in Assumption 8.1.2, we suggest verifying the required mixing condition by modeling the network state process as a function of a Poisson Point Process (PPP). PPP's are popular in stochastic geometry (Haenggi 2012) and it was recently shown by (Ramesan and Baccelli 2021) that under a mobility model for a PPP, the PPP is α-mixing. A future direction is to formulate a suitable mobility model for a PPP to guarantee α-mixing with a required decay rate. Again, another important direction is to verify α-mixing directly from data. Khaleghi and Lugosi (2023) also presented a hypothesis test to decide (using a single sample path of a stochastic process) whether the weighted sum of α-mixing coefficients of a process is finite. With this, we can verify α-mixing with a required decay rate for a network process. We envision an initialization phase for distributed interactions executed over a wireless network, where agents transmit pilot signals, and each agent locally observes its network state. Using the results from Khaleghi and Lugosi (2023), each agent can decide with high confidence whether its network state process is α-mixing with a specific rate after enough samples have been obtained. In combination with conditions for Assumption 8.1.1, the agents can then collaboratively decide whether they can solve an optimization problem in a distributed manner in a given communication environment and how to adapt their optimization stepsize.

This work thus connects a network model that is representative of practical wireless communication with an abstract optimization framework for a rich class of distributed stochastic optimization problems. The strength of the result is that it enables a class of AoI-aware MAC protocols for transmission scheduling. In the future, we seek to optimize the convergence behavior of distributed algorithms at runtime by adaptive network MAC protocols.

8.4 Proofs of Chapter 8

Lemma 8.4. *(Bradley 2005, Thm. 5.2) Suppose X and Y are independent α-mixing processes. Define a process Z by $Z_n := F_n(X_n, Y_n)$, where F_n are Borel functions. Then, Z is α-mixing with $\alpha(Z, n) \leq \alpha(X, n) + \alpha(Y, n)$.*

Proof of Lemma 8.2. Fix $(i, j) \in E$. Consider a backup scheduling step at some time step l, i.e., a time-step after the AoI-aware schedule in Algorithm 4. Recall the AoI variables associated with direct communication $\tilde{\tau}_{ji}$ as defined in Section 8.1.3. If $\tilde{\tau}_{ji}(l) > M_1$, then with probability $(1 - \delta)$ agent i picks $p_l^i = 1$, while all other agents stay silent with probability δ. Hence, by

Assumption 8.1.1 and the law of total probability, we have

$$\mathbb{P}\left(A_l^{ij} \mid \tau_{ji}(l) > M_1\right) \geq (1-\delta)\delta^{D-1}\mathbb{P}\left(g_l^{ij} > \beta\nu_l^j\right) > 0.$$

On the other hand,

$$\mathbb{P}\left(\bigcup_{k=l-M_1}^{l-1} A_{ij}^k \mid \tilde{\tau}_{ji}(l) \leq M_1\right) = 1.$$

The lemma now follows by the law of total probability and the stationarity of g_n^{ij} and ν_n^j from Assumption 8.1.2. □

Proof of Lemma 8.3. We have $\mathbb{1}_{A_n^{ij}} = \mathbb{1}_{(\beta,\infty)}(\mathrm{SINR}_n^{ij})$ from (8.8). By Lemma 8.4, it is thus enough to show that SINR_n^{ij} is α-mixing with exponential decay. Consider now the AoI-aware schedule in Algorithm 4. The AoI-aware schedule can be stated as

$$p_n^i = \mathbb{1}_{[0,\infty)}\left(\Pi_n^i - u_n^i\right),$$

with u_n^i i.i.d. uniform random variables on $[0,1]$. The representation (8.14) shows that p_n^i is a Borel function of u_n^i and the indicator functions $\mathbb{1}_{\{\tau_{ij}(n)>m\}}$ with $m \leq M_2$.

The events up to $\{\tau_{ij}(n) > M_2\}$ are generated by the events A_n^{kl} for all $k,l \in V$ with $m \in [n-M_2, n-1]$. It therefore follows that all $\mathbb{1}_{\{\tau_{ij}(n)>m\}}$ with $m \leq M_2$ are Borel functions of SINR_{kl}^m for $m \in [n-M_2, n-1]$. It thus follows from the SINR model (8.5) that the schedule p_n^i is a Borel function of Ψ^m and p_m^i for $m \in [n-M_2, n-1]$, and additional i.i.d. processes (u_n^i and time-dependent i.i.d. Borel functions in the SINR model (8.5)).

Consider a time step $n = sM$ for some $s \in \mathbb{N}$, i.e., the first step after the AoI-unaware schedule. From the last paragraph, it follows that p_n^i depends on p_m^i for $m \in [n-M_2, n-1]$, these are the AoI-unaware scheduling steps, which are either independent or deterministic and hence especially α-mixing with exponential decay. Moving forward in time, we can now see that $\forall n$ with $sM \leq n < (s+1)M$ we have that p_n^i is a Borel function of Ψ^m for $m \in [sM-M_2, n-1]$ and the decisions p_m^i in the AoI-unaware subintervals of $m \in [sM-M_2, n]$. Notably, $(\Psi^{sM-M_2}, \ldots, \Psi^{n-1})$ is also α-mixing with exponential decay, with $s \in \mathbb{N}$ such that $sM \leq n < (s+1)M$. Then, each SINR_n^{ij} is a Borel function of independent α-mixing processes with exponential decay and i.i.d. processes. Lemma 8.4 therefore shows that SINR_n^{ij} is α-mixing with exponential decay. □

Chapter 9

Numerical Experiments

This chapter presents experiments supporting the theoretical findings on multi-agent deep reinforcement learning in Chapter 4. We study the empirical convergence behavior of the proposed 3DPG algorithm in two environments and evaluate its robustness to AoI. These experiments were conducted as part of (Redder, Ramaswamy, and Karl 2022a; Redder, Ramaswamy, and Karl 2022d). As with the previous numerical examples, code and simulation data are available on `https://github.com/aredder`.

The 3DPG AC-iteration was presented in (4.9); Algorithm 5 presents the complete algorithm, including communication. Recall that each agent has a local policy parameterized by ϕ^i and a local critic parameterized by θ^i. For the experiments, we also use target networks for the local policies and critics and an Ornstein–Uhlenbeck processes $\mathcal{N}^i$ for exploration; both are described in (Lillicrap et al. 2016).

9.1 Multi-agent policy gradient for particle control

As the first experiment, we compare 3DPG, as presented in Chapter 4, and MADDPG, as presented in (Lowe et al. 2017), for a centralized training setting with global information access. Recall that 3DPG implements policy gradient updates based on the local policies of other agents, whereas MADDPG was proposed with policy gradient updates based on the local actions of other agents; from the analysis in Section 4.4.3 we expect the use of local policies to be superior at the cost of potential higher training variance.

In the second experiment, we evaluate 3DPG with communication, where local states, actions, and policies must be communicated. Recall that both 3DPG and MADDPG apply to Markov game settings as discussed in Chapter 4. But, 3DPG considers that all information is inherently local and thus has to be communicated. In this way, 3DPG is applicable to true online learning.

Algorithm 5 3DPG Algorithm at agent i

Randomly initialize critic and actor weights θ_0^i, ϕ_0^i.
Randomly initialize actor weights ϕ_0^j for all $j \neq i$.
for the entire duration **do**
    Receive current local state s_n^i.
    Execute local action $a_n^i = \mu^i(s_n^i; \phi_n^i) + \mathcal{N}_n^i$.
    Observe local rewar r_{n+1}^i and local state s_{n+1}^i.
    Allocate local data $(s_n^i, a_n^i, s_{n+1}^i)$ and current local policy ϕ_i^n for transmission to other agents.
    Run communication protocols.
    Store completely received global tuples t_m^i in the local replay memory R_n^i.
    Sample M transitions from the local replay memory R_n^i.
    Apply iteration (4.9) using the sampled transitions.

9.1.1 Environment and simulation details

For the experiments, we consider a simplified version of the simple spread multi-particle coordination problem in (Lowe et al. 2017): Agents and landmarks are represented by point masses in $[-1, 1]^2$. Moreover, agents can move around by choosing a displacement from the set $[-0.1, 0.1]^2$. Agents can observe their relative distance to the landmarks and other agents. The actual simple spread environment considers that agents and landmarks take room in space, and the agents are penalized for collisions. The described simplified setting serves the purpose of comparing the convergence properties of 3DPG and MADDPG.

Both MADDPG and 3DPG use the following algorithm configurations, chosen based on a rough hyperparameter sweep for both algorithms.

- Discount factor $\alpha = 0.9$; replay memory size 20000; minibatch size 128; two-layer GELU neural networks for each local policy with 64 and 8 neurons and tanh output layer; two-layer GELU neural networks for each local critic with 1024 and 64 neurons.
- Critic and actor stepsizes are chosen as

$$\alpha(n) = \frac{e^{-6}}{\frac{n}{1000} + 1} \quad \text{and} \quad \beta(n) = \frac{e^{-6}}{\frac{n}{1000} + 1} + \frac{e^{-6}}{(\frac{n}{1000} + 1)^2}$$

  respectively.
- The critic and policy networks are updated with the ADAM optimizer (Kingma and Ba 2015) using Tensorflow 2.

Experiment 1

In the first experiment, we validate the findings on using local policies vs. local actions in policy gradient updates in 3DPG and MADDPG, respectively. For the comparison, we add an additional coordination layer to the introduced version of the simple spread multi-particle environment (SMPE). In the SSMPE, particles (the agents) move in the plane to cover a number of landmarks, and the landmarks are episodically reset to new positions. At every time step, the agents get a global reward

$$\exp\left(-\text{average closest distance to the landmarks}\right) \in (0, 1). \tag{9.1}$$

We did not observe any significant difference between 3DPG and MADDPG in this particular scenario. This is because in SSMPE, the agents' actions do not require any form of coordination, and the global reward at every time step is only a function of the global state.

We now add a "coordination layer" task to the environment. We consider two agents that move around in the plane to minimize their average distance to three landmarks as before. However, the agents only get rewarded if they orientate in the same direction. Specifically, the previous global reward gets weighted by

$$\exp\left(-\text{angle between the orientation of the two agents}\right). \tag{9.2}$$

Consequently, the two agents only get the previous reward (9.1) when they orientate in the same direction. The optimal policy is, therefore, to spread in an optimal way to cover the 3 landmarks, while this position should be approached from a desired angle "agreed between the agents" to maximize the reward. The new reward structure, therefore, requires that the agents make more coordinated decisions. Due to the random actions taken during training, it will be difficult for the MADDPG agents to agree on a policy since the optimal path for an agent to the optimal location now depends significantly more on the path taken by the other agent.

We trained both 3DPG and MADDPG with global data access and global access to all agents' policies, i.e., for the centralized training setting without communication and AoI. We trained the agents for 10 seeds over 1500 epochs, where each epoch had a horizon of 25 steps. Figure 9.1 shows the resulting average reward per epoch. The simulations support the theoretical predictions from Chapter 4: *For problems that require coordinated actions, 3DPG obtains better policies faster than MADDPG at the cost of higher training variance.* The simulations thus reinforce the following practical suggestion: Using other agents' policies during training instead of agents' actions yields better policy improvement when a multi-agent decision-making problem requires coordinated actions. As most problems of practical interest will require coordination, we believe that using other agents' policies (potentially old versions) should be generally preferred for multi-agent actor-critic learning.

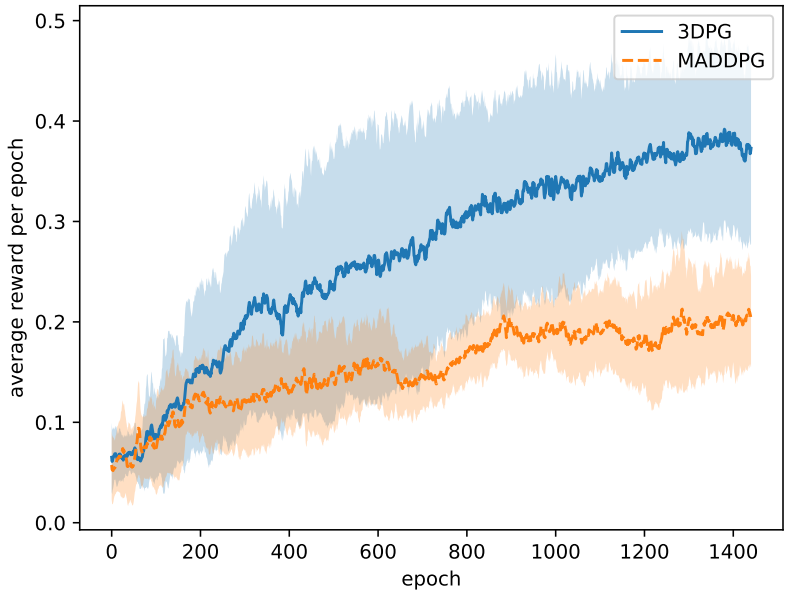

Figure 9.1: Comparison of 3DPG and MADDPG with centralized training.

Experiment 2

In the second experiment, we show that 3DPG with communication is robust to large AoI and that 3DPG may even benefit from using older policies of other agents similar to how target networks improve training in single-agent RL (Lillicrap et al. 2016).

We again consider the two agents and three landmarks problem from Experiment 1. In addition, we consider that each of the two agents uses an independent communication channel for communication. Specifically, each agent has a fixed communication budget of 15000 bits/timeslot to communicate with the other agent whenever their channel access is successful. We emulate lossy communication by varying the access probability

$$\lambda \in \{e^{-1}, e^{-2}, e^{-3}, e^{-4}\} \approx \{0.3679, 0.1353, 0.0498, 0.0183\}$$

of a simple Bernoulli access channel (Tse and Viswanath 2005). Further, we use the same hyper-parameter configurations as in Experiment 1. For the assumed communication budget, this implies that the agents require at least 3 successive successful communication events to exchange a parameter vector ϕ_n^i, while alternative at least 33 local tuples $(s_n^i, a_n^i, s_{n+1}^i)$ could be exchanged during the same time. We let each agent cycle between communicating a policy update and communicating 33 local tuples. In that sense, we give "equal weight" to policy and data communication. As a Bernoulli channel is strongly mixing with an arbitrary fast rate, it follows from Theorem 6.3 that a dominating random variable with arbitrary moment bound exists for the AoI and DataAoI random variables affecting 3DPG and Assumption 4.2.1 holds.

For experiment 2, we trained the agents for 10 seeds over 2000 epochs, where each epoch had again a horizon of 25 steps. We display the average reward per epoch in Figure 9.2. We see that 3DPG with even $\lambda = e^{-4}$ can learn decent policies compared to centralized 3DPG, albeit at a slower convergence rate. Notably, the $\lambda = e^{-4}$ run achieves this with AoIs frequently over 500 time steps (20 epochs) as shown in Figure 9.3. In addition, 3DPG with $\lambda = e^{-4}$ has only access to $1/3$ of the global data tuples that 3DPG uses with centralized training. This shows that 3DPG is highly robust to AoI and low data availability. Finally, an interesting observation is that 3DPG training runs with $\lambda = e^{-1}, e^{-2}$ or e^{-3} consistently performed similar or even better than 3DPG with centralized training. For $\lambda = e^{-1}$, we consistently observed better initial training progress, which indicates that 3DPG may benefit from using older policies of other agents in a similar manner as target networks stabilize training.

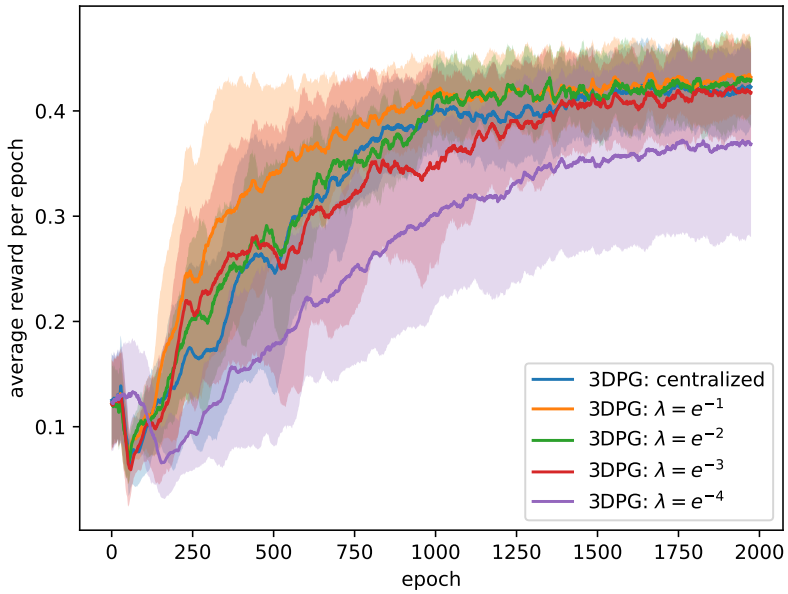

Figure 9.2: Average reward per epoch of 3DPG with variance over seeds.

9.2 Multi-agent learning for cyber-physical systems

The last experiment is based on (Redder, Ramaswamy, and Karl 2022d). The paper studied a cyber-physical two-agent flow control problem and applied 3DPG with communication over a wireless network. Flow control is an important problem in chemical industries, data center cooling, and water treatment plants. In the two-agent example, each agent must control one flow control valve. Specifically, we consider the water filter in Figure 9.4, which will be modeled as a continuous time system. In Figure 9.4, agent 1 has to control the inflow to the water filter from a main water line using valve a_1. Agent 2 has to control the outflow of the water filter

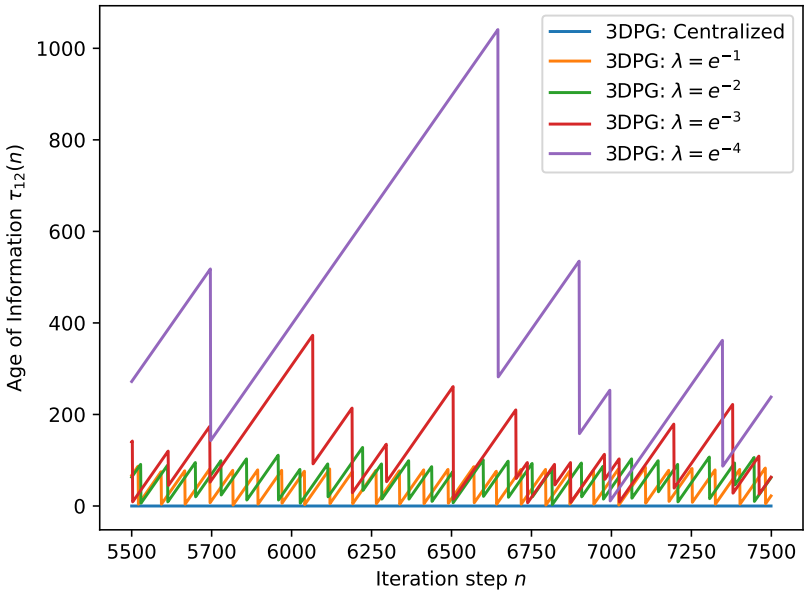

Figure 9.3: Comparison of 3DPG with communication; λ is the communication success probability.

using valve a_2. We consider that time is slotted into small intervals of constant length Δ. The discrete time steps are indexed by $n \in \mathbb{N}$. As in the previous chapter, we refer to a time slot n as the time interval from time step $n-1$ to n. For this subsection, we will use upper indices n to denote the discrete time index.

We consider that the valves can be positioned continuously from close to open for both agents. Therefore, at every time step n the agents have to pick actions $a_1^n, a_2^n \in [0,1]$ for the associated time slot n. For every time slot n, we denote the sampled flow in the main line by $s_1^n \in S_1 \subset \mathbb{R}_{\geq 0}$ and the sampled water level in the water filter by $s_2^n \in S_2 \subset \mathbb{R}_{\geq 0}$. The problem is complicated since the agents have no information about the dynamics of the main water line and the water filter. Agent 1 can only observe state s_1^n, but not state s_2^n, and vise versa for agent 2. Additionally, the agents have no information about the policy of the other agent, i.e., the agents initially have no information about the strategy of the other agent to control its valve. However, we assume that the agents can use a wireless network to exchange information, enabling the use of Algorithm 5.

Given s_1^n and a_1^n for some time step n, we assume a continuous function $f(s_1^n, a_1^n)$ that determines the inflow to the filter during time slot n. Equivalently, we can assume that agent 1 can measure the inflow to the filter. Other than that, we make no additional assumptions. *The agents have no information about the filter's dynamics or flow in the main water line.*

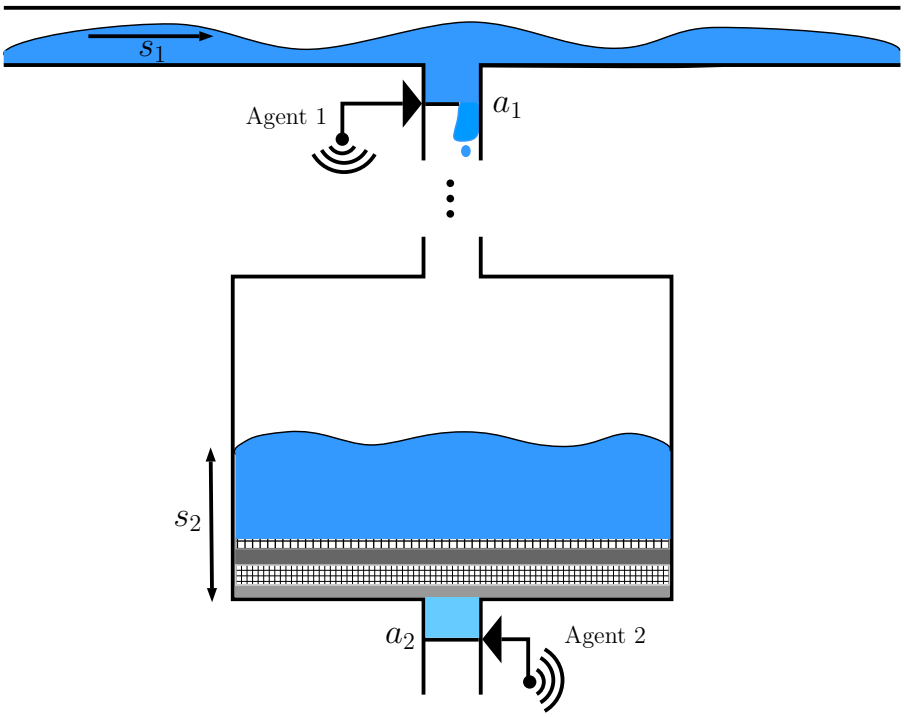

Figure 9.4: Illustration of the two-agent water filter flow control problem.

Partial Observability Issues

Algorithm 5 formulates a decentralized multi-agent solution based on locally observed states. The training procedure, however, can use delayed global information that has been communicated over a network. The solution, therefore, falls under the paradigm of decentralized training with communicated central data but decentralized execution. Depending on how the local state spaces are defined, agents may or may not be able to find good solutions for a problem. In general, the local policies should only use local states, such that inference is decentralized.

Now, consider the flow control problem. If agent 1 can only observe the state s_1 and never state s_2, it can only learn to take conservative actions since it has no information about the current amount of water in the water filter. We expect better solutions if both agents can observe the whole state space $s = (s_1, s_2)$. A system theoretic solution to this would now be to use a suitable local estimator and replace s_2^n by an estimate $\hat{s}_2^n$ at agent 1 to execute a local policy that is a function of the global state space. Alternatively, one could directly replace s_2^n with its delayed counterpart and use a recurrent architecture μ_1 and μ_2 (Hausknecht and Stone 2015). Since the objective with this example is to illustrate the effect of lossy communication on the learning algorithm and not the effect on the inference quality, we assume here for illustration that both agents can observe s_1 and s_2. Below, we formulate the MDP for the two-agent flow control problem, where both agents observe $s = (s_1, s_2)$. Moreover, the agents, therefore, only need to communicate their local actions a_1^n and a_2^n, as well as their policy parametrizations ϕ_1^n and ϕ_2^n.

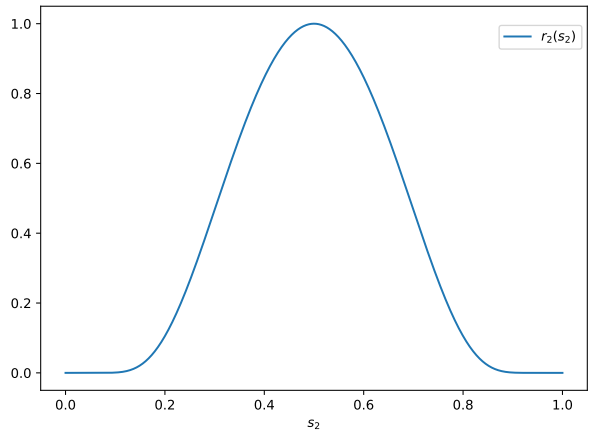

Figure 9.5: Plot of the second component, $r_2(s_2)$, of the reward function.

An MDP for Two-Agent Flow Control

We consider the following high-level objective for the two-agent flow control problem:

(i) Maximize the through flow of the filter.
(ii) Avoid under- and overflows, i.e. $0 < s_2^n < 1$ for all $n \geq 0$.
(iii) Try to keep a reserve of $s_2^n \approx 1/2$ as good as possible for all $n \geq 0$.

The state space for both agents is defined as $\mathcal{S}_{1,2} \coloneqq S_1 \times S_2$. The action spaces are defined as $\mathcal{A}_1 \coloneqq [0, 1]$ and $\mathcal{A}_2 \coloneqq [0, 1]$, respectively. To achieve the formulated objective, we consider a continuous reward function $r(s_1, s_2, a_1) = f(s_1, a_1)c + r_2(s_2)$, with

$$r_2(s_2) = \exp\left(\frac{(2s_2 - 1)^2}{(s_2 - 1)s_2}\right), \tag{9.3}$$

and some constant $c > 0$ that may be used to weight the importance of (i) over (ii) and (iii). Recall that f is the function that determines the inflow to the water filter. The inflow can be measured by agent 1 and will be communicated to agent 2 alongside s_1 and a_1, which is only required for training and not for inference. Figure 9.5 shows the second component of the reward function. We choose this function to satisfy points (ii) and (iii) of the objective. Clearly, various other shapes for the reward function are possible (Eschmann 2021).

Network Model

We merely consider two Bernoulli processes to emulate a delaying communication network as in the previous experiment. We assume that $\mathbb{P}(A_i^n) = \lambda$ at every time step n for both agent 1 and 2. We then choose $\lambda \in \{1, 1/2, 1/4, 1/8, 1/16, 1/32\}$. Practically, this corresponds to a suitable choice of a power schedule for given i.i.d. interference-noise processes. Generalizations to the more representative network model of the previous chapter are directly available but do not add

additional value here to the present simulation. However, to make the communication model more realistic, we assume a narrowband wireless channel with $B = 10Mhz$ and a typical choice $\beta = 7$ for the SINR threshold. Then, we can obtain an effective bitrate of $19^{Mbit/s}$. Now, suppose we code every real number using 64 bits. Then, observe that each policy contains 4544 real-valued parameters, and since we merely require the exchange of the actions a_i, we only need to exchange one real number to obtain a sample (s_i, a_i). Therefore, we require at least 16 time slots of successful communication to transmit a single policy update, while in each time slot we can transmit more than 300 samples. As in the previous experiments, we give roughly equal bandwidth usage to the parameter- and sample transmissions. Specifically, each agent transmits a single update to its parameter vector followed by the transmission of up to $300 \cdot 16$ samples or until its backlog is empty.

9.2.1 Simulation

We can now present the numerical evaluation of Algorithm 5 for the water flow control MDP of Section 9.2. The goal is to illustrate the convergence of 3DPG to control policies with meaningful physical behavior while the two agents communicate over a lossy communication network.

System model and algorithm details

For the simulation of the water filter system, we consider the following differential equation

$$A\frac{ds_2(t)}{dt} - Ks_1(t)a_1(t) + L(\rho g s_2(t))a_2(t) = 0 \tag{9.4}$$

as a model for the water filter. Here A is the area of the water filter, ρ is the density of water, g is the acceleration of mass, and K and L are constants. The factor $\rho g s_2(t)$ is the so-called differential pressure at time t (Rossiter 2021). The second term is a bilinear model for the inflow of the system. The third term is a bilinear model for the outflow of the filter. The constant L includes the filter's resistance and the outflow pipe's resistance. This is only an approximate model for "small" changes of the flow. However, it serves the purpose of the evaluation. For the simulation, we solve (9.4) numerically for time slots n of length $\Delta = 1s$. For the constants in (9.4), we use $K = 0.1$, which corresponds to a maximum inflow of 10% of the main flow, and $L = 10^{-5}$, which models the high resistance of the water filter. Finally, we use $c = 0.1$ to give high weight to balancing the water filter around $s_2 = 1/2$.

For Algorithm 3DPG, we use the following set of parameters: Both agents approximate a local Q-function by a two-layer neural network with 256 and 128 GELUs as activations. Two-layer neural networks are then used to represent both agent policies, where each layer has 64 GELUs and a tanh output layer. We use discount factor $\gamma = 0.95$ and a training batch size of 128.

Training and evaluation

We train the agents over 500 epochs with length $T = 100$ and use the following learning stepsize schedules

$$\alpha(n) = \frac{e^{-8}}{\frac{n}{1000} + 1}, \quad \beta(n) = \frac{e^{-8}}{\frac{n}{1000} + 1} + \frac{e^{-7} - e^{-8}}{\left(\frac{n}{1000} + 1\right)^2}. \tag{9.5}$$

As before these, learning rates satisfy the stepsize assumptions proposed for 3DPG in Chapter 4.

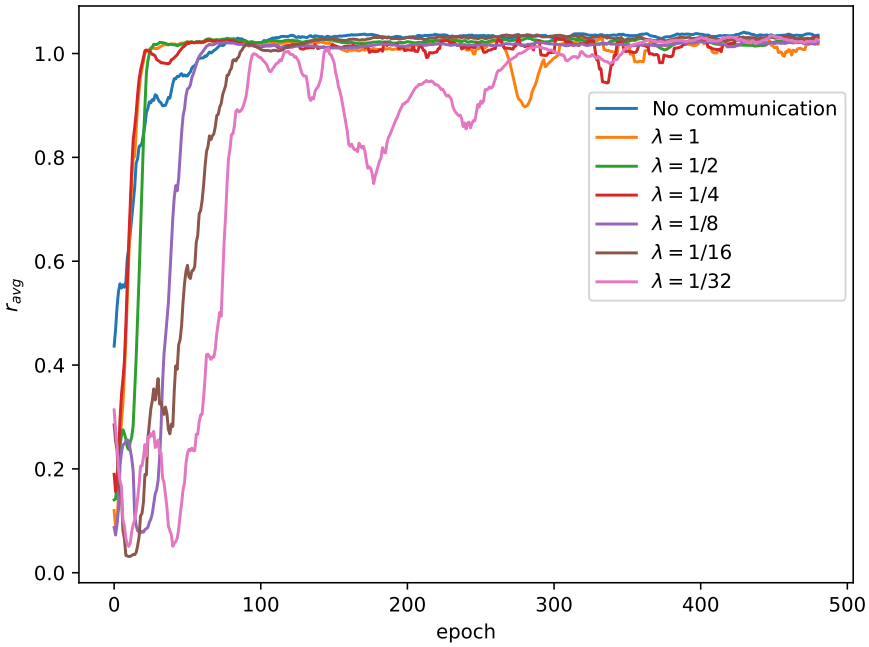

Figure 9.6: Average reward after each training epoch.

Figure 9.6 shows the average reward at the end of each training epoch evaluated without exploration noise on a new trajectory of length $T = 1000$ for every λ. Observe that for all λ the algorithm converges to a policy of similar average reward compared Algorithm 5 without communication and global information access. Moreover, as λ decreases, the convergence rate also decreases. As already observed in the previous experiments, for small values of λ the effect of lossy communication is minor.

After training, we evaluated the final policies on a new trajectory. For illustration, we show the results for $\lambda = 1/16$. In Figure 9.7, we show the inflow to the water filter, the water level in the filter, and the reward per step. In the water level plot, we see that the agents learned to quickly balance the water level around the desired height $s_2^n = 1/2$. The inflow is upper bounded by the maximal possible inflow for $a_1^n = 1$. The reward per step is upper bounded by 1 plus the aforementioned upper bound. However, the maximal possible inflow and, therefore, the maximal possible reward per step may not be obtainable since opening the inflow valve fully may increase the desired water level height even when the outflow valve is fully opened. We can observe this

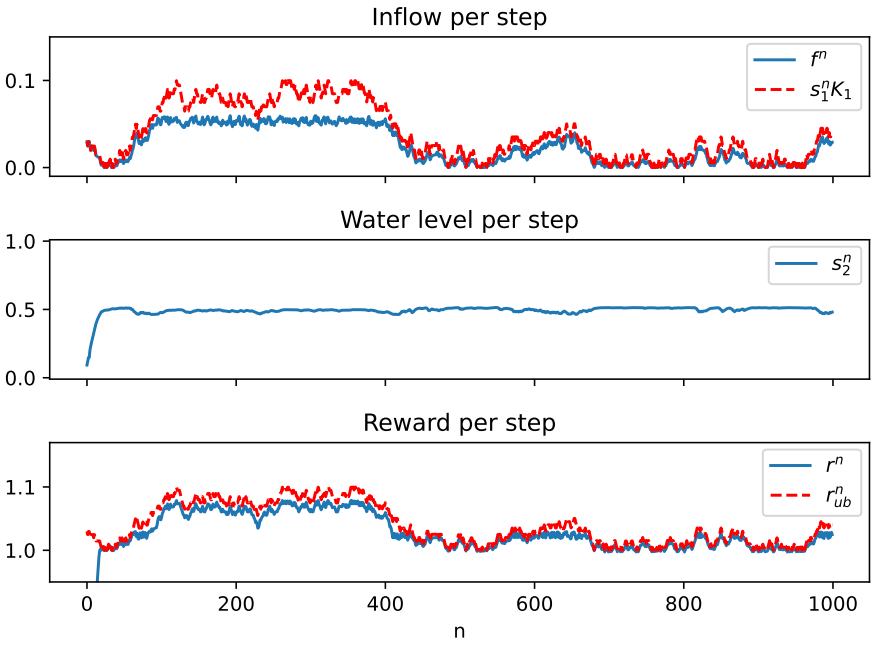

Figure 9.7: Main flow, inflow, water level, and reward per step during one trajectory after training. In the top plot, the red graph represents the maximum possible inflow. In the bottom plot, the red graph is the maximum possible reward per step.

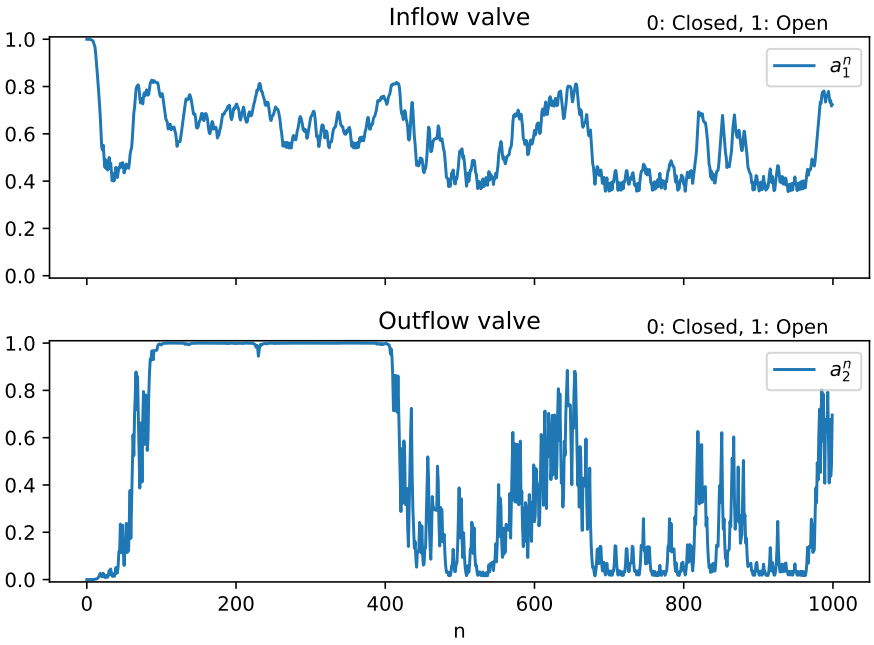

Figure 9.8: Actions associated with the simulated trajectory after training.

between time steps $n = 50$ to $n = 400$. To see this, consider the associated actions during the trajectory in Figure 9.8. We see that when the inflow is to large in $[50, 400]$, agent 1 has to reduce the inflow to the system, while agent 2 has to fully open the outflow. In this time interval, the inflow part of the reward becomes significant, and therefore, the agents learned to maximize the inflow.

We can also observe that for the second half of the trajectory, the algorithm could potentially improve with longer training. Here, the agents mainly balance the water level at the desired height since the inflow is relatively small and since we chose the weighting factor $c = 0.1$. However, opening the inflow valve more can still lead to a slightly larger reward per step. In summary, we have seen that agents can learn non-trivial policies for an unknown environment in the presence of lossy communication.

Chapter 10

Conclusions and Future Directions

This thesis investigated stability and convergence conditions for distributed asynchronous stochastic approximation and deep reinforcement learning algorithms. The proposed verifiable conditions guarantee that errors due to information delays vanish asymptotically when appropriate stepsizes are chosen. Further, the results enable accelerated learning and optimization on highly heterogeneous (volunteer) computing infrastructure (Anderson 2020; Tirmazi et al. 2020). For example, based on the results in Chapters 2, 3 and 7, we can design resource scheduling problems to optimize the assignment of heterogeneous workers to subspaces of optimization and learning problems. Beyond computing scenarios, the results enable truly online multi-agent learning based on communicated information in resource-constrained scenarios that only allow minimal communication between learning systems. Here, the combination of Chapters 4 and 8 enables verifiable conditions for mobile agents that communicate over a resource-constrained wireless network prone to potentially heavy-tailed interference (Clavier et al. 2020). We will now discuss future directions.

In Chapter 2, we generalized the BMT to distributed SA algorithms. Ramaswamy and Shalabh Bhatnagar (2017) generalized the original BMT to set-valued SA algorithms, which have many applications to approximation algorithms with errors or biases. Currently, no stability and convergence theory exists for set-valued SA algorithms with information delays. A potential Lipschitz continuity condition for a set-valued map $H : \mathbb{R}^n \to \{\text{subsets of } \mathbb{R}^m\}$ may read as follows: There exists some $L > 0$, such that

$$H(x) \subset H(y) + B_{L\|x-y\|}(0). \tag{10.1}$$

Based on a Lipschitz condition of this form, we expect to develop a BMT for set-valued SA with delays. In addition, it is also left to combine the distributed BMT with asynchronous SA (V. S. Borkar 1998). In other words, one can combine the methods used to prove the distributed BMT with the asynchronous version from Shalabh Bhatnagar (2011) or the more recent version from Yu, Wan, and R. S. Sutton (2023).

Beyond distributed optimization, we applied the BMT techniques to heavy-ball momentum SA. It will be interesting whether we can generally apply the tools to SA algorithms that approximate second-order dynamical systems, i.e., SA algorithms that asymptotically track solutions to second-order ODEs of the form

$$\frac{d^2x}{dt^2} + A(t)\frac{dx}{dt} = B(t)h(x) \tag{10.2}$$

for some drift/vector field $h(x)$ and a some matrix-valued processes $A(t), B(t)$. A stability theory for SAs that track ODEs with dynamics (10.2) will be useful to study newly designed ODE-inspired algorithms (S. Meyn 2022).

In Chapter 3, we proved a simple asymptotic, almost sure convergence rate estimate for the DASGD algorithm. However, the obtained convergence rate estimate is only available for $p > 1$, where p is some AoI moment bound. Some additional analysis is needed to complete the picture. Furthermore, it will be interesting to study DASGD in combination with adaptive stepsize schemes recently proposed for ASGD with AoI in the order of the number of workers (Mishchenko et al. 2022; Anastasiia Koloskova, Stich, and Jaggi 2022).

In Chapter 4, we studied multi-agent actor-critic reinforcement learning prone to aged communicated samples used during training. the results showed that under weak assumptions, the accumulated experience of the multi-agent system still guarantees that the state-Markov process converges to a stationary distribution. An interesting question is whether we can design data scheduling methods for multi-agent systems to enhance the convergence rate of the state-Markov process. In other words, how to design an online method that schedules a limited amount of communication resources to agents of the MAS to enhance the convergence rate of the global state-Markov process. Finally, it will be interesting to study the deep MARL framework studied in Chapter 4 with clipped gradient dynamics (Ramaswamy, Shalabh Bhatnagar, and Saxena 2023) as a two-timescale algorithm with set-valued dynamics.

In Chapter 6, we derived moment bounds for AoI processes based on strongly mixing event processes that describe when new information arrives from a source at a monitor. The natural question is whether one can generalize these results to AoI and event processes in continuous time (Brandes, Curato, and Stelzer 2023). Strong mixing naturally generalizes to continuous time, but the splitting techniques used in Chapter 6 require changes and some new ideas for the continuous-time setting.

In Chapter 7, we showed how parallel point and renewal processes naturally describe asynchronous computing scenarios. Furthermore, we derived closed-form expressions for the limiting AoI. A key assumption for this statement is the independence of the point processes that describe the processing of individual systems. But if dependent jobs are scheduled to the systems, this assumption does not generally hold. It will be a challenge to extend the results to dependent parallel point processes. However, techniques from De la Pena and Giné (2012) on decoupling

and conditional independent sequence can be a starting point. With this, we can incorporate information on the dependence of jobs arriving at a computing infrastructure into the framework.

Finally, in Chapter 8, we proposed a class of AoI-dependent network scheduling protocols that preserve the strong mixing of the underlying event process that defines an AoI process driven by Markovian dynamics. The limitations of the protocol class are that one needs to implement short time windows with an AoI-independent policy. We believe that one can completely remove the limitations at the cost of a slower mixing rate of the resulting event process with network scheduling.

Chapter A

Appendix

A.1 Analysis and dynamical systems

Theorem (Discrete Gronwall Inequality (V. Borkar 2022, Appendix B)). *Let x_n, a_n non-negative (respectively positive) sequences and $C, L > 0$ scalars such that for all n, $x_{n+1} \leq C + L\sum_{m=0}^{n} a_m x_m$, then $x_{n+1} \leq C\exp(L\sum_{m=0}^{n} a_m)$.*

Theorem (General Arzelà–Ascoli Theorem (Munkres 2000, Theorem 47.1)). *Let f_n be a sequence of functions $f_n : \mathbb{R}^d \to \mathbb{R}^k$. If the collection $\{f_n\}$ is pointwise bounded and equicontinuous, then the sequence f_n has a subsequence that converges to a continuous function in the topology of compact convergence.*

Theorem (Rademachers Theorem (Evans 2018, Sec. 3.1)). *Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be a Lipschitz-Continuous function, then f is differentiable almost everywhere, i.e. the points $x \in \mathbb{R}^n$ where f is non-differentiable form a set of Lebesque measure zero.*

Theorem (Gradient Theorem for Lipschitz-Continuous Functions). *Let $f : \mathbb{R}^n \to R$ be a Lipschitz-continuous function and $\gamma : [0,1] \to \mathbb{R}^n$ a continuously differentiable path, then*

$$\int_0^1 \nabla_x f(\gamma(t))\frac{d}{dt}\gamma(t)dx = f(\gamma(1)) - f(\gamma(0)).$$

Proof. Since $[0,1]$ is compact, $f \circ \gamma$ is Lipschitz-Continuous, hence absolutely continuous as well as continuously differentiable almost everywhere on $[0,1]$. The absolute continuity and the fundamental theorem of calculus yield that $\int_0^1 \frac{d}{dt} f(\gamma(t))dt = f(\gamma(1)) - f(\gamma(0))$. Using the multivariate chain rule and that $f \circ \gamma$ is continuously differentiable almost everywhere, the statement follows. □

A.2 Probability theory

Theorem (Law of total probability)**.** *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be probability space. Let $A_0, A_1, A_2, \ldots$ be elements of $\mathcal{F}$ that form a countably infinite partition of Ω. Then for every $E \in \mathcal{F}$,*

$$\mathbb{P}(E) = \sum_k \mathbb{P}(E \cap A_k). \tag{A.1}$$

Theorem. *For a non-negative random variable X, its expected value can be expressed as*

$$\mathbb{E}[X] = \int_0^\infty \mathbb{P}(X > x)\, dx. \tag{A.2}$$

Proposition A.1 (Law of total variance, (Ross 2014, Prop. 3.1))**.**

$$Var(X) = \mathbb{E}[Var(X \mid Y)] + Var(\mathbb{E}[X \mid Y]).$$

Theorem (First Borel-Cantelli Lemma (K. L. Chung 2001, Theorem 4.2.1))**.** *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be probability space. Let $A_0, A_1, \ldots$ be elements of $\mathcal{F}$, such that $\sum_{n=0}^\infty \mathbb{P}(A_i) < \infty$, then $\mathbb{P}(A_k \text{ i.o.}) = \mathbb{P}(\limsup A_k) = 0$.*

Theorem (Portmanteau Theorem (Billingsley 2013, Section 2))**.** *$X(n) \Rightarrow X$ if and only if*

$$\lim_{n\to\infty} \mathbb{E}[f(X(n))] = \mathbb{E}[f(X)] \tag{A.3}$$

for all bounded, continuous real functions f.

Theorem (Egorovs Theorem (Rudin 1987, p. 73))**.** *Let X_n, X be random variables. Then $X_n \to X$ almost surely if and only if $X_n \to X$ almost uniformly.*

Theorem (Martingale Convergence Theorem (V. Borkar 2022, Appendix C))**.** *Let $\{(X_n, \mathcal{F}_n)\}$ be a martingale, if $\mathbb{E}\left[\|X_n\|^2\right] < \infty$ for all $n \geq 0$, then X_n converges almost surely on the set $\sum_{n\geq 0} \mathbb{E}\left[\|X_{n+1} - X_n\|^2 \mid \mathcal{F}_n\right] < \infty$.*

Theorem (Portmanteau Theorem (Billingsley 2013, Section 2))**.** *$X(n) \Rightarrow X$ if and only if*

$$\lim_{n\to\infty} \mathbb{E}[f(X(n))] = \mathbb{E}[f(X)] \tag{A.4}$$

for all bounded, continuous real functions f.

Theorem (Blackwell's Theorem, (Serfozo 2009, Sec. 2.18 and references therein))**.** *Let $N(t)$ be a renewal process with $\mu := \mathbb{E}[W(n)] < \infty$. Then, for every fixed $t' \geq 0$,*

$$N(t) - N(t - t') \Rightarrow \tilde{N}(t'), \tag{A.5}$$

as $t \to \infty$, where $\tilde{N}(t')$ is the (stationary) modified renewal process associated with $N(t)$. Further,

$$\mathbb{E}[N(t)] - \mathbb{E}\left[N(t - t')\right] \to \frac{t'}{\mu}. \tag{A.6}$$

Usually, only (A.6) is referred to as Blackwell's Renewal Theorem, but it is an integral part of its proof to establish the weak convergence of $N(t) - N(t - t')$ to its associated modified stationary renewal process, e.g., using coupling techniques.

Bibliography

Agarwal, Alekh and Duchi, John C (2011). "Distributed delayed stochastic optimization". In: *Advances in neural information processing systems* 24.

Amos, Brandon, Xu, Lei, and Kolter, J Zico (2017). "Input convex neural networks". In: *International Conference on Machine Learning*. PMLR, pp. 146–155.

Anderson, David P (2020). "BOINC: a platform for volunteer computing". In: *Journal of Grid Computing* 18.1, pp. 99–122.

Arulkumaran, Kai et al. (2017). "Deep reinforcement learning: A brief survey". In: *IEEE Signal Processing Magazine* 34.6, pp. 26–38.

Aubin, J-P and Cellina, Arrigo (2012). *Differential inclusions: set-valued maps and viability theory*. Vol. 264. Springer Science & Business Media.

Aviv, Rotem Zamir et al. (2021). "Asynchronous distributed learning: Adapting to gradient delays without prior knowledge". In: *International Conference on Machine Learning*. PMLR, pp. 436–445.

Avrachenkov, Konstantin, Patil, Kishor, and Thoppe, Gugan (2022). "Online algorithms for estimating change rates of web pages". In: *Performance Evaluation* 153, p. 102261.

Bandi, Chaithanya, Trichakis, Nikolaos, and Vayanos, Phebe (2019). "Robust multiclass queuing theory for wait time estimation in resource allocation systems". In: *Management Science* 65.1, pp. 152–187.

Barabasi, Albert-Laszlo (2005). "The origin of bursts and heavy tails in human dynamics". In: *Nature* 435.7039, pp. 207–211.

Barakat, Anas and Bianchi, Pascal (2021). "Convergence and dynamical behavior of the ADAM algorithm for nonconvex stochastic optimization". In: *SIAM Journal on Optimization* 31.1, pp. 244–274.

Barth-Maron, Gabriel et al. (2018). "Distributed distributional deterministic policy gradients". In: *arXiv preprint arXiv:1804.08617*.

Bellman, R (1957). *Dynamic Programming*. Rand Corporation research study. Princeton University Press.

Ben-Nun, Tal and Hoefler, Torsten (2019). "Demystifying parallel and distributed deep learning: An in-depth concurrency analysis". In: *ACM Computing Surveys (CSUR)* 52.4, pp. 1–43.

Benaïm, M. (2006). "Dynamics of stochastic approximation algorithms". In: *Seminaire de probabilites XXXIII*. Springer, pp. 1–68.

Benaïm, Michel (1996). "A dynamical system approach to stochastic approximations". In: *SIAM Journal on Control and Optimization* 34.2, pp. 437–472.

Benveniste, Albert, Métivier, Michel, and Priouret, Pierre (2012). *Adaptive algorithms and stochastic approximations*. Vol. 22. Springer Science & Business Media.

Berbee, Henry (1987). "Convergence rates in the strong law for bounded mixing sequences". In: *Probability theory and related fields* 74.2, pp. 255–270.

Bernstein, Daniel S et al. (2002). "The complexity of decentralized control of Markov decision processes". In: *Mathematics of operations research* 27.4, pp. 819–840.

Bertsekas, D. P. and Tsitsiklis, J. N. (1996). *Neuro-dynamic programming*. Vol. 5. Athena Scientific Belmont, MA.

Bertsekas, Dimitri (1982). "Distributed dynamic programming". In: *IEEE transactions on Automatic Control* 27.3, pp. 610–616.

Bertsekas, Dimitri and Tsitsiklis, John (2015). *Parallel and distributed computation: numerical methods*. Athena Scientific.

Bhatia, Nam P and Szegö, George P (2006). *Dynamical systems: stability theory and applications*. Vol. 35. Springer.

Bhatnagar, S.., Prasad, HL, and Prashanth, LA (2013). *Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods*. Springer.

Bhatnagar, Shalabh (2011). "The Borkar–Meyn theorem for asynchronous stochastic approximations". In: *Systems & control letters* 60.7, pp. 472–478.

Bhatnagar, Shalabh, Borkar, Vivek S, and Guin, Soumyajit (2023). "Actor-Critic or Critic-Actor? A Tale of Two Time Scales". In: *IEEE Control Systems Letters*.

Bianchi, Giuseppe (2000). "Performance analysis of the IEEE 802.11 distributed coordination function". In: *IEEE Journal on selected areas in communications* 18.3.

Bianchi, Pascal, Fort, Gersende, and Hachem, Walid (2013). "Performance of a distributed stochastic approximation algorithm". In: *IEEE Transactions on Information Theory* 59.11, pp. 7405–7418.

Billingsley, Patrick (2008). "Probability & measure". In: *W. & S.*

— (2013). *Convergence of probability measures*. John Wiley & Sons.

Boban, Mate, Gong, Xitao, and Xu, Wen (2016). "Modeling the evolution of line-of-sight blockage for V2V channels". In: *IEEE 84th Vehicular Technology Conference*.

Borkar, V.S. (2022). *Stochastic Approximation: A Dynamical Systems Viewpoint: Second Edition*. Texts and Readings in Mathematics. Hindustan Book Agency. ISBN: 9788195196111.

Borkar, Vivek S (1998). "Asynchronous stochastic approximations". In: *SIAM Journal on Control and Optimization* 36.3, pp. 840–851.

— (2006). “Stochastic approximation with ‘controlled Markov’noise”. In: *Systems & control letters* 55.2, pp. 139–145.

Borkar, Vivek S and Konda, Vijaymohan R (1997). “The actor-critic algorithm as multi-time-scale stochastic approximation”. In: *Sadhana* 22.4, pp. 525–543.

Borkar, Vivek S and Meyn, Sean P (2000). “The ODE method for convergence of stochastic approximation and reinforcement learning”. In: *SIAM Journal on Control and Optimization* 38.2, pp. 447–469.

Bradley, Richard C (2005). “Basic properties of strong mixing conditions. A survey and some open questions”. In: *Probability surveys* 2, pp. 107–144.

Brandes, Dirk-Philip, Curato, Imma Valentina, and Stelzer, Robert (2023). “Inheritance of strong mixing and weak dependence under renewal sampling”. In: *Journal of Applied Probability* 60.2, pp. 435–451.

Bravo, Mario (2016). “An adjusted payoff-based procedure for normal form games”. In: *Mathematics of Operations Research* 41.4, pp. 1469–1483.

Brillinger, David R (1969). “The calculation of cumulants via conditioning”. In: *Annals of the Institute of Statistical Mathematics* 21, pp. 215–218.

Bubeck, Sébastien et al. (2023). “Sparks of artificial general intelligence: Early experiments with gpt-4”. In: *arXiv preprint arXiv:2303.12712*.

Canese, Lorenzo et al. (2021). “Multi-agent reinforcement learning: A review of challenges and applications”. In: *Applied Sciences* 11.11, p. 4948.

Chakraborti, Subhabrata, Jardim, Felipe, and Epprecht, Eugenio (2018). “Higher-order moments using the survival function: The alternative expectation formula”. In: *The American Statistician*.

Chandy, K Mani and Lamport, Leslie (1985). “Distributed snapshots: Determining global states of distributed systems”. In: *ACM Transactions on Computer Systems (TOCS)* 3.1, pp. 63–75.

Chowdhery, Aakanksha et al. (2022). “Palm: Scaling language modeling with pathways”. In: *arXiv preprint arXiv:2204.02311*.

Chung, Kai Lai (2001). *A course in probability theory*. Academic press.

Cinlar, Erhan and ðCınlar, E. (2011). *Probability and stochastics*. Vol. 261. Springer.

Clavier, Laurent et al. (2020). “Experimental evidence for heavy tailed interference in the IoT”. In: *IEEE Communications Letters* 25.3, pp. 692–695.

Cohen, Alon et al. (2021). “Asynchronous stochastic optimization robust to arbitrary delays”. In: *Advances in Neural Information Processing Systems* 34, pp. 9024–9035.

Conway, John B (2019). *A course in functional analysis*. Vol. 96. Springer.

Cox, David Roxbee (1962). “Renewal theory”. In: *Chapman and Hall*.

Cox, David Roxbee and Isham, Valerie (1980). “Point processes”. In: *CRC Press* 12.

Cox, David Roxbee and Smith, Walter L (1954). "On the superposition of renewal processes". In: *Biometrika* 41.1-2, pp. 91–99.

Daley, Daryl J (1978). "Bounds for the variance of certain stationary point processes". In: *Stochastic Processes and their Applications* 7.3, pp. 255–264.

Daley, DJ and Rolski, T (1992). "Finiteness of waiting-time moments in general stationary single-server queues". In: *The Annals of Applied Probability*, pp. 987–1008.

Davydov, Yu A (1974). "Mixing conditions for Markov chains". In: *Theory of Probability & Its Applications* 18.2, pp. 312–328.

De Finetti, Bruno (1970). *Teoria delle probabilità: sintesi introduttiva con appendice critica. 2 (1970)*. Einaudi.

De la Pena, Victor and Giné, Evarist (2012). *Decoupling: from dependence to independence*. Springer Science & Business Media.

Dean, Jeffrey et al. (2012). "Large scale distributed deep networks". In: *Advances in neural information processing systems* 25.

Deb, Rohan and Bhatnagar, Shalabh (2021). "N-Timescale Stochastic Approximation: Stability and Convergence". In: *arXiv preprint arXiv:2112.03515*.

— (2022). "Gradient Temporal Difference with Momentum: Stability and Convergence". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36, pp. 6488–6496.

Devraj, Adithya M, Bušić, Ana, and Meyn, Sean (2019). "On matrix momentum stochastic approximation and applications to Q-learning". In: *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, pp. 749–756.

Doob, Joseph L (1948). "Renewal theory from the point of view of the theory of probability". In: *Transactions of the American Mathematical Society* 63.3, pp. 422–438.

Durrett, Rick (2019). *Probability: Theory and Examples*. Cambridge University Press.

Eschmann, Jonas (2021). "Reward function design in reinforcement learning". In: *Reinforcement Learning Algorithms: Analysis and Applications*, pp. 25–33.

Evans, LawrenceCraig (2018). *Measure theory and fine properties of functions*. Routledge.

Farazi, Shahab, Klein, Andrew G, and Brown, D Richard (2020). "Average age of information in update systems with active sources and packet delivery errors". In: *IEEE Wireless Communications Letters* 9.8, pp. 1164–1168.

Feller, Willy (1941). "On the integral equation of renewal theory". In: *The Annals of Mathematical Statistics* 12.

Feyzmahdavian, Hamid Reza, Aytekin, Arda, and Johansson, Mikael (2016). "An asynchronous mini-batch algorithm for regularized stochastic optimization". In: *IEEE Transactions on Automatic Control* 61.12, pp. 3740–3754.

Frühwirth-Schnatter, Sylvia (2006). "Finite mixture and Markov switching models". In: *Springer*.

Gadat, Sébastien, Panloup, Fabien, and Saadane, Sofiane (2018). "Stochastic heavy ball". In: *Electronic Journal of Statistics* 12.1, pp. 461–529. DOI: 10.1214/18-EJS1395. URL: https://doi.org/10.1214/18-EJS1395.

Gasull, Armengol, López-Salcedo, José A, and Utzet, Frederic (2015). "Maxima of Gamma random variables and other Weibull-like distributions and the Lambert W W function". In: *Test* 24, pp. 714–733.

Ge, Rong et al. (2015). "Escaping from saddle points—online stochastic gradient for tensor decomposition". In: *Conference on learning theory*. PMLR, pp. 797–842.

Georgiou, Yiannis (2010). "Contributions for resource and job management in high performance computing". In: *Theses. Université de Grenoble*, p. 24.

Ghadimi, Saeed and Lan, Guanghui (2012). "Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization i: A generic algorithmic framework". In: *SIAM Journal on Optimization* 22.4, pp. 1469–1492.

Goodfellow, Ian, Bengio, Yoshua, and Courville, Aaron (2016). *Deep learning*. MIT press.

Graham, Ronald L et al. (1989). "Concrete mathematics: a foundation for computer science". In: *Computers in Physics* 3.5, pp. 106–107.

Guan, Lei et al. (2019). "XPipe: Efficient pipeline model parallelism for multi-GPU DNN training". In: *arXiv preprint arXiv:1911.04610*.

Gupta, Varun and Osogami, Takayuki (2011). "On Markov–Krein characterization of the mean waiting time in M/G/K and other queueing systems". In: *Queueing Systems* 68, pp. 339–352.

Haan, Laurens and Ferreira, Ana (2006). "Extreme value theory: an introduction". In: *Springer*.

Haenggi, Martin (2012). *Stochastic geometry for wireless networks*. Cambridge University Press.

Harold, J, Kushner, G, and Yin, George (1997). "Stochastic approximation and recursive algorithm and applications". In: *Application of Mathematics* 35.

Hausknecht, Matthew and Stone, Peter (2015). "Deep recurrent q-learning for partially observable mdps". In: *2015 aaai fall symposium series*.

He, Shaoming et al. (2020). "Distributed estimation over a low-cost sensor network: A review of state-of-the-art". In: *Information Fusion* 54, pp. 21–43.

Hendrycks, Dan and Gimpel, Kevin (2016). "Gaussian error linear units (gelus)". In: *arXiv preprint arXiv:1606.08415*.

Hinterstoisser, Stefan et al. (2018). "On pre-trained image features and synthetic images for deep learning". In: *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*.

Huang, Yanping et al. (2019). "Gpipe: Efficient training of giant neural networks using pipeline parallelism". In: *Advances in neural information processing systems* 32.

Iksanov, Alexander (2016). *Elements of renewal theory, with applications*.

Iyer, A, Rosenberg, C, and Karnik, A (May 2009). "What is the right model for wireless channel interference?" In: *IEEE Trans. Wireless Commun.* 8.5, pp. 2662–2671.

Jin, Chi et al. (2017). "How to escape saddle points efficiently". In: *International conference on machine learning*. PMLR, pp. 1724–1732.

Khaleghi, Azadeh and Lugosi, Gábor (2023). "Inferring the mixing properties of a stationary ergodic process from a single sample-path". In: *IEEE Transactions on Information Theory*.

Khazaei, Hamzeh, Misic, Jelena, and Misic, Vojislav B (2011). "Performance analysis of cloud computing centers using m/g/m/m+ r queuing systems". In: *IEEE Transactions on parallel and distributed systems* 23.5, pp. 936–943.

Kiefer, Jack and Wolfowitz, Jacob (1952). "Stochastic estimation of the maximum of a regression function". In: *The Annals of Mathematical Statistics*, pp. 462–466.

Kingma, Diederik P and Ba, Jimmy (2015). "Adam: A method for stochastic optimization". In: *ICLR*.

Kingman, John FC (2009). "The first Erlang century—and the next". In: *Queueing systems* 63.1-4, p. 3.

Koloskova, Anastasia et al. (2020). "A unified theory of decentralized SGD with changing topology and local updates". In: *International Conference on Machine Learning*. PMLR, pp. 5381–5393.

Koloskova, Anastasiia, Stich, Sebastian U, and Jaggi, Martin (2022). "Sharper convergence guarantees for asynchronous sgd for distributed and federated learning". In: *Advances in Neural Information Processing Systems* 35, pp. 17202–17215.

Kombrink, Sabrina (2018). "Renewal theorems for processes with dependent interarrival times". In: *Advances in Applied Probability* 50.4, pp. 1193–1216.

Krizhevsky, Alex, Hinton, Geoffrey, et al. (2009). "Learning multiple layers of features from tiny images". In.

Kuczma, Marek (2009). "Subadditive Functions". In: *An Introduction to the Theory of Functional Equations and Inequalities: Cauchy's Equation and Jensen's Inequality*, pp. 455–481.

Kumar, Harshat, Koppel, Alec, and Ribeiro, Alejandro (2019). "On the sample complexity of actor-critic method for reinforcement learning with function approximation". In: *arXiv preprint arXiv:1910.08412*.

Kunze, Lars et al. (2018). "Artificial intelligence for long-term robot autonomy: A survey". In: *IEEE Robotics and Automation Letters* 3.4, pp. 4023–4030.

Lakshminarayanan, Chandrashekar and Bhatnagar, Shalabh (2017). "A stability criterion for two timescale stochastic approximation schemes". In: *Automatica* 79, pp. 108–114.

Lei, Jinlong et al. (2020). "On synchronous, asynchronous, and randomized best-response schemes for stochastic Nash games". In: *Mathematics of Operations Research* 45.1, pp. 157–190.

Leskelä, Lasse and Vihola, Matti (2013). "Stochastic order characterization of uniform integrability and tightness". In: *Statistics & Probability Letters* 83.1, pp. 382–389.

Lian, Xiangru et al. (2015). "Asynchronous parallel stochastic gradient for nonconvex optimization". In: *Advances in neural information processing systems* 28.

Lillicrap, Timothy P et al. (2016). "Continuous control with deep reinforcement learning." In: *ICLR (Poster)*.

Lim, Hyojun and Kim, Chongkwon (2001). "Flooding in wireless ad hoc networks". In: *Computer Communications* 24.3-4, pp. 353–363.

Lin, Siyu et al. (2015). "Finite-state Markov modeling for high-speed railway fading channels". In: *IEEE Antennas and Wireless Propagation Letters* 14, pp. 954–957.

Littman, Michael L (1994). "Markov games as a framework for multi-agent reinforcement learning". In: *Machine learning proceedings 1994*. Elsevier, pp. 157–163.

Liu, Ji and Wright, Stephen J (2015). "Asynchronous stochastic coordinate descent: Parallelism and convergence properties". In: *SIAM Journal on Optimization* 25.1, pp. 351–376.

Liu, Jun and Yuan, Ye (2022). "On almost sure convergence rates of stochastic gradient methods". In: *Conference on Learning Theory*. PMLR, pp. 2963–2983.

Liu, Wei and Shi, Peng (2019). "Optimal linear filtering for networked control systems with time-correlated fading channels". In: *Automatica* 101, pp. 345–353.

Liu, Yanli, Gao, Yuan, and Yin, Wotao (2020). "An improved analysis of stochastic gradient descent with momentum". In: *Advances in Neural Information Processing Systems* 33, pp. 18261–18271.

Ljung, Lennart (1977). "Analysis of recursive stochastic algorithms". In: *IEEE transactions on automatic control* 22.4, pp. 551–575.

Lowe, Ryan et al. (2017). "Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments". In: *Advances in Neural Information Processing Systems* 30, pp. 6379–6390.

McLeish, Don L (1975). "A maximal inequality and dependent strong laws". In: *The Annals of probability* 3.

McLeod, Robert M (1965). "Mean value theorems for vector valued functions". In: *Proceedings of the Edinburgh Mathematical Society* 14.3, pp. 197–209.

Mertikopoulos, Panayotis et al. (2020). "On the almost sure convergence of stochastic gradient descent in non-convex problems". In: *Advances in Neural Information Processing Systems* 33, pp. 1117–1128.

Meyn, Sean (2022). *Control systems and reinforcement learning*. Cambridge University Press.

Meyn, Sean P and Tweedie, Richard L (2012). *Markov chains and stochastic stability*. Springer Science & Business Media.

Mishchenko, Konstantin et al. (2022). "Asynchronous sgd beats minibatch sgd under arbitrary delays". In: *Advances in Neural Information Processing Systems* 35, pp. 420–433.

Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David, et al. (2015). "Human-level control through deep reinforcement learning". In: *Nature* 518.7540.

Montanari, Andrea and Saeed, Basil N (2022). "Universality of empirical risk minimization". In: *Conference on Learning Theory*. PMLR, pp. 4310–4312.

Montero, Miquel and Villarroel, Javier (2016). "Directed random walk with random restarts: The Sisyphus random walk". In: *Physical Review E* 94.3, p. 032132.

Mou, Wenlong et al. (2020). "On linear stochastic approximation: Fine-grained Polyak-Ruppert and non-asymptotic concentration". In: *Conference on Learning Theory*. PMLR, pp. 2947–2997.

Munkres, J.R. (2000). *Topology Second Edition*. Pearson.

Narayanan, Deepak et al. (2019). "PipeDream: Generalized pipeline parallelism for DNN training". In: *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, pp. 1–15.

Nesterov, Yurii (2003). *Introductory lectures on convex optimization: A basic course*. Vol. 87. Springer Science & Business Media.

Nesterov, Yurii Evgen'evich (1983). "A method of solving a convex programming problem with convergence rate $\mathcal{O}(1/k^2)$". In: *Doklady Akademii Nauk*. Vol. 269. Russian Academy of Sciences, pp. 543–547.

Netrapalli, Praneeth (2019). "Stochastic gradient descent and its variants in machine learning". In: *Journal of the Indian Institute of Science* 99.2, pp. 201–213.

Nota, Chris and Thomas, Philip S (2019). "Is the policy gradient a gradient?" In: *arXiv preprint arXiv:1906.07073*.

Pham, Tuan D and Tran, Lanh T (1985). "Some mixing properties of time series models". In: *Stochastic processes and their applications* 19.2, pp. 297–303.

Pollard, David (2002). *A user's guide to measure theoretic probability*. 8. Cambridge University Press.

Polyak, Boris T (1964). "Some methods of speeding up the convergence of iteration methods". In: *Ussr computational mathematics and mathematical physics* 4.5, pp. 1–17.

Prasad, HL, LA, Prashanth, and Bhatnagar, Shalabh (2015). "Two-timescale algorithms for learning Nash equilibria in general-sum stochastic games". In: *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pp. 1371–1379.

Raina, Rajat, Madhavan, Anand, and Ng, Andrew Y (2009). "Large-scale deep unsupervised learning using graphics processors". In: *Proceedings of the 26th annual international conference on machine learning*, pp. 873–880.

Ramachandran, Prajit, Zoph, Barret, and Le, Quoc V (2017). "Searching for activation functions". In: *arXiv preprint arXiv:1710.05941*.

Ramaswamy, Arunselvan and Bhatnagar, Shalabh (2017). "A generalization of the Borkar-Meyn theorem for stochastic recursive inclusions". In: *Mathematics of Operations Research* 42.3, pp. 648–661.

Ramaswamy, Arunselvan, Bhatnagar, Shalabh, and Quevedo, Daniel E (2020). "Asynchronous stochastic approximations with asymptotically biased errors and deep multiagent learning". In: *IEEE Transactions on Automatic Control* 66.9, pp. 3969–3983.

Ramaswamy, Arunselvan, Bhatnagar, Shalabh, and Saxena, Naman (2023). "A Framework for Provably Stable and Consistent Training of Deep Feedforward Networks". In: *arXiv preprint arXiv:2305.12125*.

Ramaswamy, Arunselvan and Hullermeier, Eyke (2021). "Deep Q-Learning: Theoretical Insights from an Asymptotic Analysis". In: *IEEE Transactions on Artificial Intelligence*. DOI: 10.1109/TAI.2021.3111142.

Ramaswamy, Arunselvan, Redder, Adrian, and Quevedo, Daniel E (2021a). "Distributed optimization over time-varying networks with stochastic information delays". In: *IEEE Transactions on Automatic Control*.

— (2021b). "Distributed optimization over time-varying networks with stochastic information delays". In: *IEEE Transactions on Automatic Control*. DOI: 10.1109/TAC.2021.3108492.

Ramesan, Nithin S and Baccelli, François (2021). "How wireless queues benefit from motion: an analysis of the continuum between zero and infinite mobility". In: *IEEE Transactions on Wireless Communications* 20.12, pp. 8149–8162.

Redder, Adrian (2023). "Age-of-Information in Distributed Systems caused by Asynchronous Computing Modeled as Parallel Renewal Processes". In: *https://doi.org/10.21203/rs.3.rs-3502945/v2*.

Redder, Adrian, Ramaswamy, Arunselvan, and Karl, Holger (2022a). "3DPG: Distributed deep deterministic policy gradient algorithms for networked multi-agent systems". In: *arXiv preprint (under review)*. arXiv: 2201.00570. URL: https://arxiv.org/abs/2201.00570.

— (2022b). "Age of Information Process under Strongly Mixing Communication-Moment Bound, Mixing Rate and Strong Law". In: *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, pp. 1–8.

— (2022c). "Asymptotic Convergence of Deep Multi-Agent Actor-Critic Algorithms". In: *To appear*.

— (2022d). "Multi-agent Policy Gradient Algorithms for Cyber-physical Systems with Lossy Communication." In: *ICAART (1)*, pp. 282–289.

— (2022e). "Practical Network Conditions for the Convergence of Distributed Optimization". In: *IFAC Conf. on Networked Systems*.

— (2023). "Stability and Convergence of Distributed Stochastic Approximations with large Unbounded Stochastic Information Delays". In: *arXiv preprint arXiv:2305.07091*.

Robbins, Herbert and Monro, Sutton (1951). "A stochastic approximation method". In: *The annals of mathematical statistics*, pp. 400–407.

Ross, Sheldon M (2014). "Introduction to probability models". In: *Academic press*.

Rossiter, Anthony (2021). *Modelling, dynamics and control*. https://controleducation.group.shef.ac.uk/chaptermodelling.html. Accessed: 25.05.2021.

Rudin, Walter (1987). *Real and Complex Analysis, 3rd Ed.* USA: McGraw-Hill, Inc. ISBN: 0070542341.

Salzman, Oren and Stern, Roni (2020). "Research challenges and opportunities in multi-agent path finding and multi-agent pickup and delivery problems". In: *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1711–1715.

Samorodnitsky, Gennady et al. (2016). *Stochastic processes and long range dependence*. Vol. 26. Springer.

Samsi, Siddharth et al. (2021). "The mit supercloud dataset". In: *2021 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE, pp. 1–8.

Scutari, Gesualdo and Sun, Ying (2019). "Distributed nonconvex constrained optimization over time-varying digraphs". en. In: *Math. Program.* 176.1-2, pp. 497–544.

Sebbouh, Othmane, Gower, Robert M, and Defazio, Aaron (2021). "Almost sure convergence rates for stochastic gradient descent and stochastic heavy ball". In: *Conference on Learning Theory*. PMLR, pp. 3935–3971.

Serfozo, Richard (2009). "Basics of applied stochastic processes". In: *Springer Science & Business Media*.

Shanechi, Maryam M et al. (2012). "Feedback-controlled parallel point process filter for estimation of goal-directed movements from neural signals". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 21.1, pp. 129–140.

Silver, David et al. (2014). "Deterministic policy gradient algorithms". In: *International conference on machine learning*. PMLR, pp. 387–395.

Sofla, Maryam Sheikh et al. (2022). "Towards effective offloading mechanisms in fog computing". In: *Multimedia Tools and Applications*, p. 1.

Su, Weijie, Boyd, Stephen, and Candes, Emmanuel J (2016). "A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights". In: *Journal of Machine Learning Research* 17.153, pp. 1–43.

Sun, Yin et al. (2019). "Age of information: A new metric for information freshness". In: *Synthesis Lectures on Communication Networks* 12.2, pp. 1–224.

Sutskever, Ilya et al. (2013). "On the importance of initialization and momentum in deep learning". In: *International conference on machine learning*. PMLR, pp. 1139–1147.

Taga, Yasushi (1963). "On high order moments of the number of renewals". In: *Annals of the Institute of Statistical Mathematics* 15, pp. 187–196.

Tagliabue, Andrea et al. (2023). "REAL: Resilience and Adaptation using Large Language Models on Autonomous Aerial Robots". In: *arXiv preprint arXiv:2311.01403*.

Tirmazi, Muhammad et al. (2020). "Borg: the next generation". In: *Proceedings of the fifteenth European conference on computer systems*, pp. 1–14.

Tse, David and Viswanath, Pramod (2005). *Fundamentals of wireless communication*. Cambridge university press.

Tsitsiklis, John, Bertsekas, Dimitri, and Athans, Michael (1986). "Distributed asynchronous deterministic and stochastic gradient optimization algorithms". In: *IEEE transactions on automatic control* 31.9, pp. 803–812.

Tsitsiklis, John N (1994). "Asynchronous stochastic approximation and Q-learning". In: *Machine learning* 16, pp. 185–202.

Uryasev, Stanislav and Pardalos, Panos M (2013). *Stochastic optimization: algorithms and applications*. Vol. 54. Springer Science & Business Media.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems* 30.

Viswanathan, Harish and Mogensen, Preben E (2020). "Communications in the 6G era". In: *IEEE Access* 8, pp. 57063–57074.

Vlaski, Stefan and Sayed, Ali H (2021). "Second-order guarantees of stochastic gradient descent in non-convex optimization". In: *IEEE Transactions on Automatic Control*.

Wang, Hong Shen and Moayeri, Nader (1995). "Finite-state Markov channel-a useful model for radio communication channels". In: *IEEE transactions on vehicular technology* 44.1, pp. 163–171.

Wang, Lingxiao et al. (2019). "Neural policy gradient methods: Global optimality and rates of convergence". In: *arXiv preprint arXiv:1909.01150*.

Wilson, Ashia C et al. (2017). "The marginal value of adaptive gradient methods in machine learning". In: *Advances in neural information processing systems* 30.

Yang, Bowen et al. (2021). "Pipemare: Asynchronous pipeline parallel dnn training". In: *Proceedings of Machine Learning and Systems* 3, pp. 269–296.

Yates, Roy D et al. (2021). "Age of information: An introduction and survey". In: *IEEE Journal on Selected Areas in Communications* 39.5, pp. 1183–1210.

Yu, Huizhen, Wan, Yi, and Sutton, Richard S (2023). "A Note on Stability in Asynchronous Stochastic Approximation without Communication Delays". In: *arXiv preprint arXiv:2312.15091*.

Zhang, Chen et al. (2021). "A survey on federated learning". In: *Knowledge-Based Systems* 216, p. 106775.

Zhang, Kaiqing et al. (2018). "Fully decentralized multi-agent reinforcement learning with networked agents". In: *International Conference on Machine Learning*. PMLR, pp. 5872–5881.

Zhang, Shanshan et al. (2013). "Asynchronous stochastic gradient descent for DNN training". In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, pp. 6660–6663.

Zhang, Xin, Liu, Jia, and Zhu, Zhengyuan (2020). "Taming convergence for asynchronous stochastic gradient descent with unbounded delay in non-convex learning". In: *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, pp. 3580–3585.

Zhang, Yizhou et al. (2023). "Global convergence of localized policy iteration in networked multi-agent reinforcement learning". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 7.1, pp. 1–51.

Zhou, Zhengyuan et al. (2022). "Distributed stochastic optimization with large delays". In: *Mathematics of Operations Research* 47.3, pp. 2082–2111.

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

City, Date:

Adrian Redder