ERROR BOUNDS AND APPLICATIONS FOR STOCHASTIC APPROXIMATION WITH NON-DECAYING GAIN

by
Jingyi Zhu

A dissertation submitted to The Johns Hopkins University
in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland
February, 2020

Abstract

This work analyzes the stochastic approximation algorithm with non-decaying gains as applied in time-varying problems. The setting is to minimize a sequence of scalar-valued loss functions $f_k(\cdot)$ at sampling times τ_k or to locate the root of a sequence of vector-valued functions $\boldsymbol{g}_k(\cdot)$ at τ_k with respect to a parameter $\theta \in \mathbb{R}^p$. The available information is the noise-corrupted observation(s) of either $f_k(\cdot)$ or $\boldsymbol{g}_k(\cdot)$ evaluated at one or two design points only. Given the time-varying stochastic approximation setup, we apply stochastic approximation algorithms. The gain has to be bounded away from zero so that the recursive estimate denoted as $\hat{\theta}_k$ can maintain its momentum in tracking the time-varying optimum denoted as θ_k^*. Given that $\{\theta_k^*\}$ is perpetually varying, the best property that $\hat{\theta}_k$ can have is to be near the solution θ_k^* (concentration behavior) in place of the improbable convergence.

Chapter 3 provides a bound for the root-mean-squared error $\sqrt{\mathbb{E}(\|\hat{\theta}_k - \theta_k^*\|^2)}$ and a bound for the mean-absolute-deviation $\mathbb{E}\|\hat{\theta}_k - \theta_k^*\|$. Note that the only assumption imposed on $\{\theta_k^*\}$ is that the average distance between two consecutive underlying optimal parameter vectors is bounded from above. Overall, the bounds are applicable under a mild assumption on the time-varying drift and a modest restriction on the observation noise and the bias term.

ABSTRACT

After establishing the tracking capability in Chapter 3, we also discuss the concentration behavior of $\hat{\theta}_k$ in Chapter 4. The weak convergence limit of the continuous interpolation of $\hat{\theta}_k$ is shown to follow the trajectory of a non-autonomous ordinary differential equation. Then we apply the formula for variation of parameters to derive a computable upper-bound for the probability that $\hat{\theta}_k$ deviates from θ_k^* beyond a certain threshold. Both Chapter 3 and Chapter 4 are probabilistic arguments and may not provide much guidance on the gain-tuning strategies useful for one single experiment run. Therefore, Chapter 5 discusses a data-dependent gain-tuning strategy based on estimating the Hessian information and the noise level. Overall, this work answers the questions "what is the estimate for the dynamical system θ_k^*" and "how much we can trust $\hat{\theta}_k$ as an estimate for θ_k^*."

Index Terms—stochastic approximation, non-decaying gain, constant gain, error bound, time-varying systems, ODE limit, second-order algorithms

Primary Reader and Academic Advisor: Dr. James C. Spall

Second Reader: Dr. Nicolas Charon

Dedication to my family.

Acknowledgments

I would like to express my sincere gratitude to my academic advisor, Dr. James Spall, for his continuing support throughout my graduate studies in the Department of Applied Mathematics and Statistics (AMS) at the Johns Hopkins University (JHU). His admirable work ethic, deep insight in the field of optimization and control, and kindness to his students, were influential in making my JHU experience fruitful and enriching.

I would like to thank Dr. Nicolas Charon for taking his time in reviewing the thesis as a second reader, and for serving on the committee of the three key oral exams in my graduate study. I also would like to thank Dr. Donniell Fishkind, Dr. Benjamin Hobbs, Dr. Enrique Mallada, Dr. Daniel Robinson, Dr. Howard Weinert, all the members of the committee of my Candidacy exam, my Graduate Board Oral exam and my dissertation defense.

I want to thank Dr. Avanti Athreya, Dr. Agostino Capponi, Dr. Nicolas Charon, Dr. Donniell Fishkind, Dr. James Fill, Dr. Vince Lyzinski, Dr. Mauro Maggioni, Dr. Carey Priebe, Dr. Daniel Robinson, Dr. James Spall, Dr. Minh Tang, Dr. Fred Torcaso, for the helpful courses they've designed and delivered in the Department of AMS. I want to thank Ms. Kristin Bechtel and Ms. Heather Kelm for their kindness and patience in coordinating

the academic-related issues. I would like to thank Ms. Denise Link-Farajali from the Center for Leadership Education at JHU and Ms. Kathy Ceasar-Spall for helping me in reviewing the grammatical issues and providing rephrasing suggestions.

Last but not least, I would like to thank my husband, Long Wang, for his friendship and love, and for the enlightening discussions we've shared. The work in Appendix A could not have been completed timely without his dedicated collaboration. I would like to thank my siblings, Jingwen Zhu and Siyi Huang, for their continuing support and trust. I would like to thank my parents, Jincheng Zhu and Qiaoming Huang, for everything they have offered my siblings and me throughout our lives.

This work was supported by the Paul V. Renoff Fellowship from JHU, the Charles and Catherine Counselman Fellowship from the Department of AMS, the Acheson J. Duncan Fund for the Advancement of Research in Statistics from the Department of AMS, the Office of Naval Research via Navy contract N00024-13-D6400, and Dr. James Spall's JHU/APL sabbatical professorship at Whiting School of Engineering in JHU.

Contents

CONTENTS

List of Tables

List of Figures

List of Notation and Acronyms

The frequently used notation are arranged by category in the following lists.

General Math Notation

Let A and B denote some sets[1], and let x and y denote some canonical points.

$\mathbb{R}^p$	The p-dimensional Euclidean space
$\mathbb{N}$	The set of natural numbers $\{0, 1, 2, \cdots\}$
$A \subset B$	Set A is a subset of set B, including the scenario where $A = B$
$A \subsetneq B$	Set A is a proper subset of set B, i.e., the possibility of $A = B$ is excluded
$A \setminus B$	The set obtained by excluding all elements in B from the set A
$\mathrm{int}(A)$	The interior of set A
∂A	The boundary of set A
$\overline{A}$	The closure of set A
$\mathrm{Ball}_\varepsilon(A)$	The ε-neighborhood of set A, defined as $\{x : \|x - y\| \leq \varepsilon, \text{for } y \in A\}$
$\mathscr{P}_A(x)$	The projection of point x onto set A, defined as $\mathrm{argmin}_{y \in A} \|x - y\|$

[1] Usually, the set of our interest in this thesis is a subset of the p-dimensional Euclidean space.

$\mathbb{I}_A(x)$	The indicator function of the set A, which gives 1 if $x \in A$ and 0 if $x \notin A$

Let $\boldsymbol{A}$ denote a matrix, and let $\boldsymbol{x}$ denote a vector. $\boldsymbol{A}$ is not necessarily square unless specified otherwise.

$\|\cdot\|$	The Euclidean norm if the input argument is a vector, or the spectral norm if the input argument is a matrix
$\|\cdot\|$	The determinant of the input matrix in $\mathbb{R}^{p\times p}$
$\boldsymbol{I}_p$	The identity matrix in $\mathbb{R}^{p\times p}$
$\lambda_{\min}(\boldsymbol{A})$	The smallest eigenvalue of $\boldsymbol{A} \in \mathbb{R}^{p\times p}$
$\lambda_{\max}(\boldsymbol{A})$	The largest eigenvalue of $\boldsymbol{A} \in \mathbb{R}^{p\times p}$
$\sigma_{\min}(\boldsymbol{A})$	The smallest singular value of $\boldsymbol{A}$
$\sigma_{\max}(\boldsymbol{A})$	The largest singular value of $\boldsymbol{A}$
$\mathrm{tr}(\boldsymbol{A})$	The trace of $\boldsymbol{A}$
$\mathrm{cond}(\boldsymbol{A})$	The condition number of $\boldsymbol{A} \in \mathbb{R}^{p\times p}$
$\mathrm{rank}(\boldsymbol{A})$	The rank of $\boldsymbol{A}$
$\mathrm{vec}\,(\boldsymbol{A})$	The vectorization of $\boldsymbol{A}$, i.e., the concatenation of all columns of $\boldsymbol{A}$
$\boldsymbol{A}^T$	The transpose of matrix $\boldsymbol{A}$
$\boldsymbol{A}_1 \succ \boldsymbol{A}_2$	The binary operator $\succ$ means that $(\boldsymbol{A}_1 - \boldsymbol{A}_2)$ is a positive definite matrix for $\boldsymbol{A}_1, \boldsymbol{A}_2 \in \mathbb{R}^{p\times p}$
$\boldsymbol{A}_1 \succeq \boldsymbol{A}_2$	The binary operator $\succeq$ means that $(\boldsymbol{A}_1 - \boldsymbol{A}_2)$ is a positive semi-definite matrix for $\boldsymbol{A}_1, \boldsymbol{A}_2 \in \mathbb{R}^{p\times p}$

$\mathrm{diag}(\boldsymbol{x})$	The matrix with diagonal entries being the components of $\boldsymbol{x}$
$\boldsymbol{x}^{-1}$	The component-wise inverse of vector $\boldsymbol{x} \in \mathbb{R}^p$ with nonzero components
$\boldsymbol{x}^{-T}$	The transpose of $\boldsymbol{x}^{-1}$ for $\boldsymbol{x} \in \mathbb{R}^p$ with nonzero components

Let $(\Omega, \mathcal{A}, \mathbb{P})$ be the probability space of interest. Let ω denote an outcome within the sample space Ω, and let $\boldsymbol{x}$ and $\boldsymbol{y}$ denote some random vectors.

ω	An elementary event belonging to the set Ω
$\mathbb{P}(A)$	The probability of a set $A \in \mathcal{A}$
$\mathbb{E}(\boldsymbol{x})$	The expectation of $\boldsymbol{x}$
$\mathbb{V}(\boldsymbol{x})$	The variance (matrix) of $\boldsymbol{x}$
$\mathbb{C}(\boldsymbol{x}, \boldsymbol{y})$	The covariance (matrix) of $\boldsymbol{x}$ and $\boldsymbol{y}$ (which is not necessarily square)
$(\mathcal{F}_i)_{i \in I}$	It is a filtration if $\mathcal{F}_i$ is a sub sigma-algebra of $\mathcal{A}$ for all i, and $\mathcal{F}_j \subset \mathcal{F}_k \subset \mathcal{A}$ for all $j \leq k$ in the index set I (which is usually $\mathbb{N}$)
$\xrightarrow{\text{dist}}$	Converges in distribution

Notation For Stochastic Optimization Framework Under Nonstationary Scenarios

The following notation are consistently used throughout the entire thesis, except that Section 2.1 and Appendix A consider the traditional stochastic approximation setting with a fixed loss function and a fixed optimum.

θ	The vector of parameters being estimated, which usually lives in $\mathbb{R}^p$
τ_k	The sampling time at the kth iteration

$f_k(\theta)$	The loss function to minimize, which varies with the index k
$\boldsymbol{g}_k(\theta)$	The gradient of the scalar-valued loss function under the minimization setting or the underlying vector-valued function in the root-finding setting
$\boldsymbol{H}_k(\theta)$	The Hessian of the scalar-valued loss function $f_k(\theta)$ under the minimization setting or the Jacobian of the underlying vector-valued function $\boldsymbol{g}_k(\theta)$ under the root-finding setting
θ_k^*	The minimizer of the scalar-valued loss function under the minimization setting or the root of the vector-valued function under the root-finding setting; the subscript k is used to emphasize that the optimum value changes with time τ_k
$\hat{\theta}_k$	The recursive estimate for θ_k^* produced in the kth iteration of an algorithm searching
$\hat{\boldsymbol{g}}_k(\theta)$	The estimator for $\boldsymbol{g}_k(\theta)$ obtained at time τ_k, which will be used in the iterative SA scheme
$\beta_k(\theta)$ and $\xi_k(\theta)$	The bias term and the error term of using $\hat{\boldsymbol{g}}_k(\theta)$ as an estimator for $\boldsymbol{g}_k(\theta)$, see page 17
$\boldsymbol{e}_k(\theta)$	The error term that is the sum of $\beta_k(\theta)$ and $\xi_k(\theta)$ (alternatively, the difference between $\hat{\boldsymbol{g}}_k(\theta)$ and $\boldsymbol{g}_k(\theta)$), see page 44
a_k or a	The *non-decaying* or *constant* positive gain appearing in the generic form of SA algorithm, see page 16

c_k	The *non-decaying* positive differencing magnitude arising in constructing $\hat{\boldsymbol{g}}_k(\cdot)$ using FDSA/SPSA schemes, see page 43
$\boldsymbol{\Delta}_k$	The perturbation vector arising from constructing $\hat{\boldsymbol{g}}_k(\cdot)$ using SPSA scheme, see page 18
$C^j(\mathbb{R} \mapsto \mathbb{R}^p)$	The set of continuous functions that map $\mathbb{R}$ to $\mathbb{R}^p$ and are jth-order[2] continuous, see page 26
$D(\mathbb{R} \mapsto \mathbb{R}^p)$	The set of functions that map from $\mathbb{R}$ to $\mathbb{R}^p$ and are right-continuous with left-hand limits, see page 33
$\mathscr{M}_k$	The square-root of the second-moment of $\boldsymbol{e}_k$, see page 47
$\mathscr{B}_k$	The square-root of the expected Euclidean-distance-squared between θ^*_{k+1} and θ^*_k, see page 47
$\mathscr{C}_k$	The convexity parameter of $\boldsymbol{g}_k(\cdot)$, see page 47
$\mathscr{L}_k$	The Lipschitz parameter of $\boldsymbol{g}_k(\cdot)$, see page 47
$\mathscr{R}_k$	The ratio between $\mathscr{L}_k$ and $\mathscr{C}_k$, see page 47

Acronyms

Below are the acronyms that will be used in the thesis.

SO	stochastic optimization, see page 1

[2]If a function that maps $\mathbb{R}$ to $\mathbb{R}^p$ is in $C^0(\mathbb{R} \mapsto \mathbb{R}^p)$, it means that it is continuous for $j \in \mathbb{N}$. If a function that maps $\mathbb{R}$ to $\mathbb{R}^p$ is in $C^1(\mathbb{R} \mapsto \mathbb{R}^p)$, it means that both its first-order derivative and itself are continuous. In general, if a function that maps $\mathbb{R}$ to $\mathbb{R}^p$ is in $C^j(\mathbb{R} \mapsto \mathbb{R}^p)$, it means that itself and its ith-order derivatives are continuous for all $i \leq j$.

LIST OF NOTATION AND ACRONYMS

SA	stochastic approximation, see page 1
SGD	stochastic gradient descent, see page 18
SG	stochastic gradient, see page 18
FD	finite difference, see page 224
FDSA	finite difference stochastic approximation, see page 11
SP	simultaneous perturbation, see page 12
SPSA	simultaneous perturbation stochastic approximation, see page 11
SP2	simultaneous perturbation with two-measurements, see page 18
SP1	simultaneous perturbation with one-measurement, see page 43
2SPSA	second-order simultaneous perturbation stochastic approximation, see page 222
2SG	second-order stochastic gradient, see page 222
E2SPSA	enhanced second-order simultaneous perturbation stochastic approximation, see page 226
E2SG	enhanced second-order stochastic gradient, see page 226
ODE	ordinary differential equation, see page 8
IVP	initial value problem, see page 25
RMS	root-mean-squared, see page 9
MAD	mean-absolute-deviation, see page 9
MSE	mean-squared-error, see page 22
LMS	least-mean-squares, see page 6

RLS	recursive-least-squares, see page 6
ERF	empirical risk function, see page 51
KF	Kalman filter, see page 8
MISO	multiple-input-single-output, see page 91
MIMO	multiple-input-multiple-output, see page 95
FLOP	floating-point-operations, see page 223
UAV/UUV	unmanned aerial/undersea vehicle, see page 1
i.i.d.	independently and identically distributed
a.s.	almost surely
m.s.	mean-squared
w.p.1.	with probability one
w.l.o.g.	without loss of generality
w.r.t.	with respect to
r.h.s.	right-hand-side
l.h.s.	left-hand-side
a.k.a.	also known as

Chapter 1

Introduction

This thesis focuses on a general stochastic optimization (SO) framework in the context of solving a nonstationary problem. In a nutshell, we are allowed to gather *noisy* zeroth- or first-order information *only*, to minimize a sequence of *time-varying* scalar-valued loss functions or locate the root for a sequence of *time-varying* vector-valued functions. The consideration of both the *randomness* and *dynamics* makes the work here different from both the traditional deterministic optimization and the classical SO.

The motivation for the time-varying SO setup comes from modern needs in areas such as electrical power distribution, unmanned aerial or undersea vehicle (UAV or UUV) tracking, and multi-agent problems. To successfully track a time-varying parameter has long been an important topic in these real-world problems, and both the time variation and the noise corruption have been notorious barriers that obstruct experimenters from making accurate statistical inference. Recursive stochastic approximation (SA) algorithms

with non-decaying gain sequence are widely used for tracking purposes. Interestingly, the theoretical foundation for the application of SA on the time-varying parameter identification is still developing, and so is guidance to tune the non-decaying gain sequence. This thesis is devoted to these issues.

Before we move forward to the technical issues, let us further motivate why we use recursive SA schemes with non-decaying gains to handle time-varying problems in Section 1.1 and explain the challenges arising from time-varying SO set up in Section 1.2. Then Section 1.3 lists the contribution of this work along with an overview of the upcoming chapters.

1.1 Motivation

Though the asymptotic properties of SA schemes with decaying gains have been studied mainly based on a fixed optimizer, SA algorithms with non-decaying gains have been widely applied in sequential processing and online learning, where true loss functions and, therefore, underlying optimums are drifting over time. Consider a parameter identification problem in estimating a sequence of time-varying unknown parameter vectors denoted as θ. In adaptive tracking, system control, and many estimation applications, the optimal value(s) of θ at the sampling time τ_k (corresponding to the index k), denoted as θ_k^*, may change over time due to the intrinsic evolution of the underlying system. When the sequence θ_k^* varies with time, SA algorithms with non-diminishing gain may be applied such that the iterative

output from SA algorithm, denoted as $\hat{\theta}_k$, tracks the time-varying system parameters θ_k^* at sampling points $\tau_1, \tau_2, \cdots$.

1.1.1 Why Time-Varying SA Problems Are Useful?

Stationary systems with "$\theta_k^* = \theta^*$ at every sampling time τ_k" are well studied in statistics and signal processing, and a voluminous literature is available describing efficient estimation and identification of θ. However, in adaptive control of complex processes and many other real-world systems, the underlying parameters reflecting the system characteristics θ_k^* at time τ_k intrinsically vary. Dynamic modeling is widely used in areas such as computer vision, macro- or micro-economic modeling, feedback control systems, UAV/UUV tracking, mobile communication, radar or sonar surveillance systems, and so on. In addition to the consideration of the time variability in θ_k^*, it is also necessary to take the randomness of the noise term. This is straightforward as noises arise in almost any case where physical system measurements or computer simulations are used to approximate a physical process.

Let us briefly discuss an example (to appear in Chapter 6) whose modeling should embrace both the time variability of the loss function and the randomness of the noise. Consider the scenario where an agent attempts to track the coordinate of a target submarine (or aircraft) based upon the sensor (e.g., radar) readings from its surroundings. By "tracking" we mean that the agent has to stay close to the target submarine, close enough

such that the target submarine is still detectable[1] by its sensor reading. The "time variability" comes in to play as the coordinates of both the target submarine and the agent itself are constantly moving. The "randomness" is also involved as the sensor readings are corrupted by observational noise. In this real-time estimation problem, the data (sensor readings) are collected "on the fly" as the submarine is operating. The reading arrives in concert with the estimation of the location coordinates of the target submarine.

In addition to the submarine tracking example, the environment changes over time in other practical settings. In the ever-changing environment, the solution (set) may vary with time too. Such a formulation is obvious in dynamic problems such as building control systems, where the optimum may change continuously. Although less obvious, the time-varying problem also arises in settings that may appear at first glance to be static. For example, an optimum financial plan for a business or a family depends on the external environment, which, of course, changes over time. In some search and optimization problems, the algorithm will be explicitly designed to adapt to a changing environment (e.g., a controlled system) and automatically provide a new estimate at the optimal value. In other cases, one needs to restart the process and find a new solution. In either sense, the problem solving may never stop.

Overall, real-world problem-solving is often difficult, two common issues of which are the measurement noise and the lack of stationarity in the solution as a result of the conditions of the problem changing over time. These two characteristics—the randomness

[1]There exists a range outside of which the sensor can no longer detect the target. See [57] for further details.

in noises and the time variation of the loss functions—will be the central topic of this thesis. Nonetheless, other challenges will not be dealt with here, including the curse of dimensionality, saddle point(s), local optimum (versus global optimum), the problem-specific constraints, and so on.

1.1.2 Why Recursive Algorithms Are Preferred?

Often, recursive algorithms (with non-decaying gain) are used in time-varying SA problems. Let $\hat{\theta}_k$ represent an estimate for θ based on k data pairs (input and corresponding output). We seek a way of computing $\hat{\theta}_{k+1}$ as the $(k+1)$-th data pair arrives. Such schemes are preferred for: (1) their ability to reveal the most recent information, (2) their low per-iteration memory and computation requirement, and (3) their fit for online processing.

First of all, the recursive form clearly exhibits the value of a new data point. As discussed in [106, Sect. 3.3], the *instantaneous* gradient is important in time-varying systems where the aim is to estimate a sequence of θ_k^*. In such systems, it is important to place more emphasis on the more recent information. Moreover, the recursive estimates $\hat{\theta}_{k+1}$ are based on combining the current estimate $\hat{\theta}_k$ with the $(k+1)$-th data pair in an efficient way. Granted, the computational advantages alone are becoming less important with the advances in computing power. However, for the *real-time* applications with the flood of data accumulation, processing a bulk of data altogether from scratch after each data acquisition may be virtually impractical. The sequential processing may be done to

reduce the computational burden (versus the batch processing of all data) or to expose the unique impact of each datum.

Let us take a *time-invariant* system for example: computing the (batch) ordinary least-squares solution in basic linear regression is prohibitive (due to the expensive matrix inversion) and whence the least-mean-squares (LMS) and recursive-least-squares (RLS) algorithms (see [106, Chap. 3] for further details). This stress upon online (instantaneous in real-time) processing of the incoming information regarding the system characteristics, such as estimation[2] and prediction, becomes more obvious when dealing with *dynamical* systems, and consequently the typical procedure is to compute a new estimate each time when a new measurement becomes available. Such a process corresponds to *recursive identification*, where a new estimate is computed at every sampling interval when we need to optimize a control system online.

Aside from the benefits brought by the recursive form, we also note one other key advantage of using SA algorithms. Several time-varying problems for adaptively tracking complex systems can be formulated as nonstationary extremal problems of a probabilistic nature. Under such a setting, direct SA methods that do not depend on the underlying probabilistic distributions are very useful.

Further, we mention that a non-decaying gain should be used to track the *time-varying* target θ_k^* in the recursive form of SA algorithms. Often, the non-decaying gain coefficient a_k is set to a constant $a > 0$ for its simplicity. Nonetheless, the tuning of a is necessary

[2]It is sometimes termed as "filtering" in engineering literature.

to provide sufficient impetus for $\hat{\theta}_k$ to keep up with the time-varying θ_k^*. A large a helps $\hat{\theta}_k$ to converge more promptly to the vicinity of θ_k^*, but a small a helps the iterates to avoid instability and divergence. Let us also note that the constant-gain algorithms are also frequently used in neural network training even when dealing with a *time-invariant* θ^* due to its robustness, even though the constant-gain iterates $\hat{\theta}_k$ will not formally converge.

In short, the efficient extraction of the dynamical properties of signals and systems in a recursive form is central in system identification. Recursive estimates are useful to adapt themselves to the system dynamics.

1.2 Challenges

The classical SA results on the convergence and the rate of convergence, which are developed based on a fixed and unique optimizer, cannot be directly transferable to the time-varying setting. There are several lingering concerns for applying SA recursions with non-decaying gains to the time-varying stochastic optimization setting.

1.2.1 Dynamic Modeling

New questions arise if a setting of interests departs from the "time-independent/stationary loss function $f(\cdot)$" and the "fixed unique optimizer θ^*." The first question is: how to wisely characterize the temporal changes in $\{\theta_k^*\}$ such that the class of loss function $\{f_k(\cdot)\}$ is sufficiently rich to embrace a class of practical scenarios and the tracking performance remains mathematically tractable?

Many existing works hinge on a known model for the target parameter evolution, including an ordinary differential equation (ODE) model or a random-walk model. For example, Kalman filtering (KF) requires a linear state equation for the $\{\theta_k^*\}$ sequence and a sequence of loss functions $\{f_k(\cdot)\}$ in the quadratic form centering at θ_k^*. The particle filter requires the conditional probabilities of θ_k^* to be *known*, though allowing nonlinearity in the underlying state-space model.

Still, the study in tracking the time-varying parameter continues because the imposed model may be invalid or is easily misspecified. It is of practical interest to circumvent the restrictive assumptions imposed on $\{\theta_k^*\}$ or the stringent requirements on the underlying loss function sequence, denoted as $\{f_k(\cdot)\}$. We hope to set up a more general perspective in that we require neither a specified linear or nonlinear evolution for $\{\theta_k^*\}$ nor the conditional probabilities regarding θ_k^*.

In the upcoming chapters, we consider the "slowly" time-varying target in the sense that the average distance between successive optimizers is strictly bounded from above. Such an assumption also includes the case where the moving target may change abruptly—the change may have a drastic magnitude shift as long as it occurs sporadically.

1.2.2 Tracking Criteria

In addition to the issues of time-varying assumptions, we also care about such a question: what properties can $\hat{\theta}_k$ possibly have when the underlying parameters θ_k^* are time-varying?

In real-world applications such as adaptive control in power-grid scheduling and time-varying communication channels [42], the optimal value of the underlying parameter is perpetually varying, so there is no convergence per se of either θ_k^* or the estimate $\hat{\theta}_k$. That is, we cannot achieve the usual notion of convergence such that $\|\hat{\theta}_k - \theta_k^*\|$ is arbitrarily small in a certain statistical sense unless the evolution law of θ_k^* is revealed to the agent. The best we could hope for is that $\hat{\theta}_k$ stays within a neighborhood of θ_k^* with a high probability, which may also be termed as "convergence to a stationary distribution" or "concentration."

Regardless of the model assumptions and corresponding algorithms, understanding the tracking error is critical to the usefulness of the resulting estimates. Chapter 3 centers on the tracking performance in terms of controlled root-mean-squared (RMS) error $\sqrt{\mathbb{E}(\|\hat{\theta}_k - \theta_k^*\|^2)}$ or a mean-absolute-deviation (MAD) error $\mathbb{E}\|\hat{\theta}_k - \theta_k^*\|$ that is bounded uniformly across k, and Chapter 4 discusses the concentration behavior in terms of a probabilistic bound.

1.2.3 Gain Tuning

Another key issue is the question of tuning the non-diminishing gain to balance tracking accuracy and stability, under the circumstance that we have no a priori information regarding the possible disturbance acting on the system. In tracking problems, the gain must be strictly bounded away from zero. Particularly, it is well known that whilst the use of a small (constant) gain decreases the magnitude of the fluctuations in $\hat{\theta}_k$, it also

decreases the ability to track the variations in θ_k^*. A larger gain enables the resulting estimate $\hat{\theta}_k$ to approach promptly to the vicinity of θ_k^*, yet it may jeopardize the tracking stability. Chapter 5 tries to provide some practical guidance on gain selection based on this compromise.

1.3 Overview of Contents and Our Contribution

In Chapter 3, Section 3.1 sets up the time-varying SA framework, Section 3.2 motivates the model assumptions, Section 3.3 derives a computable error bound for general SA algorithms with non-decaying gains to be applied in parameter estimation along with the supporting numerical examples, and Section 3.4 lists some examples for applications. In short, this chapter illustrates that the bound is favorably informative under reasonable assumptions on the evolution of the true parameter being estimated. Specifically, the tracking capability established in Chapter 3 differs from prior literature on error bound analysis in the following senses:

(1) The restrictions placed on the model of the time-varying parameter is mild compared to the other assumed forms of the state equation. The only imposed assumption is that the average distance between two consecutive underlying parameters θ_k^* is strictly bounded from above. This modest assumption does not eliminate jumps in the target, and also allows the target to vary stochastically.

(2) Biased estimators of the gradient information may be used in SA algorithms, whereas most prior works are on unbiased estimators. With this extension, the tracking capability for a broad class of SA algorithms, including simultaneous perturbation stochastic approximation (SPSA) and finite difference stochastic approximation (FDSA), is established.

(3) Many prior works are developed with a constant gain that is tuned in advance for successful tracking and claim that the tracking error can be made smaller by decreasing the constant gain. Our discussion reveals that the adaptive gain selection should depend on the shape of the loss function, the noise level, and the drift level. Furthermore, the gain should be neither too large nor too small.

(4) The computable bound applies to general nonlinear SA algorithms with non-decaying gain and is valid for the entire time. Based on this, we can characterize the tracking performance of a large class of SA algorithms in response to the drift by determining the allowable region for the non-decaying gain sequence. Also, finite-sample analysis is possible, as the bound is not based on the vanishing gain and associated limit theorems.

Overall, our setup applies to a general scenario that allows unbounded noise, a biased gradient estimator, a drift (the dynamics being tracked) term without any explicit evolution model, and is useful for both finite-sample and asymptotic analysis.

In Chapter 4, Section 4.1 studies the weak convergence limit of the constrained SA algorithms applied in tracking time variation, and Section 4.2 quantifies the concentration behavior of the constant-gain stochastic gradient descent (SGD) algorithm within finite iterations in terms of a computable probabilistic error bound. The concentration behavior discussed in Chapter 4 differs from other works on limiting behavior in terms of the several subtleties which are further discussed in Section 4.3. Section 4.1 develops the main result of characterizing the recursive iterates via the trajectory of a nonautonomous ODE with the same initialization under proper time scaling. The bound in Section 4.2 is non-asymptotic as it is derived for the actual constant gain and not from an idealized limiting scenario based on a limit theorem for fluctuations as the constant gain goes to zero, which is often the case in prior studies. This is useful because we are considering the problem of continuously tracking a time-varying target. Also, our derivation of the bound reveals its dependence on relevant parameters, desired accuracy, and problem dimension.

Chapter 5 provides gain-tuning guidance based upon the observable information. In addition to the general gain selection strategy to ensure a bounded MAD in Chapter 3, we also develop data-dependent methods to test if an abrupt jump arises and the corresponding strategy to tune the gain sequence adaptively according to the observed information. As the Hessian and the observation error information needed to carry out jump detection are unknown, we employ the simultaneous perturbation (SP) method to estimate them. In the numerical simulation, we implement the SGD algorithm. Results support that our data-dependent gain tuning strategy helps detect abrupt changes. Note that many

prior works are on the constant gain by assuming that the gain is tuned[3] for successful tracking, whereas we discuss data-dependent gain-tuning strategy. Furthermore, the adaptive step-size scheme for constrained (truncated) stochastic approximation algorithms is useful in dynamic environments where the underlying parameters are time-varying.

Finally, Chapter 6 presents a problem that fits the time-varying SA problem setup and numerically illustrates the SA schemes non-decaying gain. Appendix A discusses a strategy to reduce the per-iteration cost of the second-order SA algorithms from $O(p^3)$ to $O(p^2)$ using the symmetric indefinite matrix factorization.

[3] A successful constant gain is highly problem-dependent.

Chapter 2

Preliminaries

This chapter lays the groundwork for upcoming discussions. Section 2.1 discusses the SO framework and presents the general form of SA algorithms (2.1).

2.1 Overview of Stochastic Approximation Algorithms

This section focuses on SA algorithms for SO via nonlinear root-finding. There are two main SO settings of interest: one is to minimize a scalar-valued function $f(\cdot)$ using its noisy evaluation $y(\cdot) \equiv f(\cdot) + \varepsilon(\cdot)$ evaluated at a certain design point θ, and the other is to locate the root(s) of a vector-valued function $\boldsymbol{g}(\cdot)$ using its corrupted observation $\boldsymbol{Y}(\cdot) \equiv \boldsymbol{g}(\cdot) + \boldsymbol{\xi}(\cdot)$ collected at a certain point θ. Here θ is the underlying parameter vector, a collection of adjustables; θ typically falls within the Euclidean p-space $\mathbb{R}^p$. Let us also denote θ^* as the (assumed unique) minimizer of the scalar-valued function $f(\cdot)$ or the (assumed unique) root of the vector-valued function $\boldsymbol{g}(\cdot)$.

The above SO settings are distinguished from deterministic optimization in that neither the direct evaluation of the scalar-valued function $f(\cdot)$ nor the exact observation of the vector-valued function $\boldsymbol{g}(\cdot)$ is available. Sometimes, the randomness in the SO process may be due to the random choice (injected randomness) made in the search direction as the algorithm iterates towards a solution to avoid getting stuck. Such scenarios commonly arise in practice. Consider a complex stochastic model whose output depends on a set of parameters θ, where the experimenter attempts to locate the value of θ that minimizes the expected output of the model. We are, under a majority of circumstances, unable to obtain a closed-form expression or exact representation for the black-box model. When dealing with physical processes in actual implementations, computing the expected value of the output for any given value of θ may be impossible in general since the physical processes are governed by rules unknown to the experimenters. While deterministic optimization techniques cannot be directly transferable to noisy environments, SO algorithms can utilize corrupted measurements to generate iterative estimates, denoted by $\hat{\theta}_k$, at each discrete-time instance k—hence the term "stochastic optimization."

There are many SO algorithms: random search (such as stochastic ruler, stochastic comparison, simulated annealing), SA (such as SGD, SPSA), and so on. The focus of this thesis is on SA. SA includes a wide range of recursive schemes (i.e., step-by-step computational methods) with decaying gain (i.e., the step-size approaches to zero as the iteration number increases) that iteratively generate $\hat{\theta}_k$ as an estimate for θ^* using the information up to the index k.

For example, given a vector-valued function $\boldsymbol{g}(\cdot) : \mathbb{R}^p \mapsto \mathbb{R}^p$, the basic SA algorithm for nonlinear root-finding aims to find the root(s) of the function $\boldsymbol{g}(\cdot)$ using the following recursive scheme:

$$\hat{\boldsymbol{\theta}}_{k+1} = \hat{\boldsymbol{\theta}}_k - a_k \hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k), \ k \in \mathbb{N}, \tag{2.1}$$

where $\{a_k\}$ is a positive gain sequence, and $\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ is the corrupted observation of the vector-valued function $\boldsymbol{g}(\cdot)$ evaluated at $\hat{\boldsymbol{\theta}}_k$. The details in constructing $\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ will be discussed in the upcoming subsection. A useful application is immediate by letting $\boldsymbol{g}(\boldsymbol{\theta}) = \partial f(\boldsymbol{\theta})/\partial \boldsymbol{\theta}$ when the iterative updating scheme (2.1) is used for SO via nonlinear root-finding. One of the caveats is that the roots of the gradient equation may not be the (global) minimizer of $f(\boldsymbol{\theta})$.

Under certain statistical or engineering conditions, $\hat{\boldsymbol{\theta}}_k$ converges a.s. or in m.s. sense to the optimum point $\boldsymbol{\theta}^*$ as $k \to \infty$ and at a certain stochastic rate. See [106, Chap. 4] for further details. Given its algorithmic robustness and computational simplicity, the recursion (2.1) with decaying gain $a_k = O(1/k)$ was pursued with great zeal by statisticians and electrical engineers as a convenient paradigm for recursive algorithms for regression, system identification, adaptive control, and so on. The subject has received a fresh lease of life in recent years because of some new emerging application areas broadly covered under the general rubric of learning[1] algorithms.

[1]They encompass learning algorithms for neural networks, reinforcement learning algorithms arising from artificial intelligence and adaptive control and models of learning by boundedly rational agents in macroeconomics.

2.1.1 General Discussion

The basic SA algorithm for nonlinear root-finding is known as the Robbins-Monro (R-M) algorithm [91]. Given a vector-valued function $\boldsymbol{g}(\cdot)$, the R-M algorithm aims to find a root of $\boldsymbol{g}(\cdot)$ recursively through (2.1), where $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ can decomposed as:

$$\begin{aligned}\hat{\boldsymbol{g}}_k(\hat{\theta}_k) &= \boldsymbol{g}(\hat{\theta}_k) + \mathbb{E}[\hat{\boldsymbol{g}}_k(\hat{\theta}_k) - \boldsymbol{g}(\hat{\theta}_k)\,|\mathcal{F}_k] + \left\{\hat{\boldsymbol{g}}_k(\hat{\theta}_k) - \mathbb{E}[\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\,|\mathcal{F}_k]\right\} \\ &\equiv \boldsymbol{g}(\hat{\theta}_k) + \boldsymbol{\beta}_k(\hat{\theta}_k) + \boldsymbol{\xi}_k(\hat{\theta}_k), \quad \text{for } k \in \mathbb{N}, \end{aligned} \tag{2.2}$$

with $\mathcal{F}_k$ being some representation of the process history. One common choice is to let $\mathcal{F}_k$ be the sigma-algebra induced by the observed quantities up until (excluding) index k. Specifically,

$$\mathcal{F}_0 = \sigma\{\hat{\theta}_0\}, \text{ and } \mathcal{F}_k = \sigma\{\hat{\theta}_0, \hat{\boldsymbol{g}}_i(\hat{\theta}_i), i < k\} \text{ for } k \geq 1. \tag{2.3}$$

If the process history is represented as in (2.3), then the l.h.s. of (2.2) is $\mathcal{F}_k$-measurable. Moreover, we can deem $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ as an estimator (in the statistical sense) of $\boldsymbol{g}_k(\hat{\theta}_k)$ at fixed point $\hat{\theta}_k$ [12]. Under such perspective, $\boldsymbol{\beta}_k(\hat{\theta}_k)$ represents the bias of $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ as an estimator of $\boldsymbol{g}(\hat{\theta}_k)$, and $\boldsymbol{\xi}_k(\hat{\theta}_k)$ is termed as the noise and is usually assumed to be a martingale difference sequence. Note that the decomposition in (2.2) is presented mainly for the purpose of analysis; in practice, the bias and noise terms are never explicitly computed

or collected. The convergence theory for the scheme (2.1) with a general form of (2.2) can be found in [106, Chaps. 4–7].

SA algorithms are often categorized according to the available information. The *zeroth-order SA* includes FDSA [55], SPSA with two-measurements [101], SPSA with one-measurement [103], RDSA [31], and so on. The *first-order SA* includes SGD and many popular machine learning algorithms. The *second-order SA* will be thoroughly reviewed in Appendix A. Two important SA algorithms are reviewed here: SPSA and the stochastic gradient (SG) form of SA.

2.1.2 Simultaneous Perturbation SA

SPSA algorithm uses *zeroth-order* information and is especially useful in the minimization setting. The recursive update for SPSA estimates is (2.1), except that $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ is substituted by $\hat{\boldsymbol{g}}_k^{\mathrm{SP2}}(\hat{\theta}_k)$ as below:

$$\hat{\boldsymbol{g}}_k^{\mathrm{SP2}}(\theta) = \frac{y(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k) - y(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)}{2c_k}\boldsymbol{\Delta}_k^{-1}, \quad \text{for } k \in \mathbb{N}, \tag{2.4}$$

where the mean-zero p-dimensional random perturbation vector $\boldsymbol{\Delta}_k$ has a user-specified distribution satisfying conditions [106, Sect. 7.3], c_k is a positive scalar governing the differencing magnitude, and $\boldsymbol{\Delta}_k^{-1}$ denotes the random vector whose individual component is the inverse of the corresponding component in $\boldsymbol{\Delta}_k$. $\mathrm{SP2}$ in the superscript is short for "simultaneous perturbation with two-measurements" [106, Sect. 7.3].

2.1.3 Stochastic Gradient Descent

The SGD algorithm uses *first-order* information and is a foundational method when an optimization problem is converted to a root-finding problem. It requires the availability of a random vector $\hat{\boldsymbol{g}}_k^{\mathrm{SG}}(\cdot)$ such that $\mathbb{E}[\hat{\boldsymbol{g}}_k^{\mathrm{SG}}(\theta)\big|\, \theta = \hat{\theta}_k] = \boldsymbol{g}(\hat{\theta}_k)$. The recursion defining $\hat{\theta}_k$ is the same as (2.1), except that generic $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ is replaced by $\hat{\boldsymbol{g}}_k^{\mathrm{SG}}(\hat{\theta}_k)$:

$$\hat{\boldsymbol{g}}_k^{\mathrm{SG}}(\hat{\theta}_k) = \boldsymbol{g}(\hat{\theta}_k) + \boldsymbol{\xi}_k^{\mathrm{SG}}(\hat{\theta}_k), \text{ with } \mathbb{E}[\boldsymbol{\xi}_k^{\mathrm{SG}}(\hat{\theta}_k)\,|\mathcal{F}_k] = \mathbf{0}, \quad \text{for } k \in \mathbb{N}. \tag{2.5}$$

The SGD algorithm is a special case of the R-M algorithm (2.1)–(2.2), as $\hat{\boldsymbol{g}}_k^{\mathrm{SG}}(\hat{\theta}_k)$ is an unbiased estimate of $\boldsymbol{g}(\hat{\theta}_k)$. Sometimes, $\hat{\boldsymbol{g}}_k^{\mathrm{SG}}(\hat{\theta}_k)$ can be obtained via *deliberate* injection of mean-zero noise, to avoid being "stuck" at a local solution.

Through the connection between root-finding and optimization, SA algorithms, such as SG (2.5) and SPSA (2.4), can be used for SO.

2.2 Review on Adaptive Tracking Algorithms

Section 2.1 discusses SA algorithms in locating an assumed unique θ^* that remains the same along the entire horizon over which we carry out the optimization procedure. However, in engineering applications about adaptive tracking or system control, the optimum solution to the underlying parameter estimation problem generally changes. The optimal values of the model parameter may change over time because of the intrinsic evolution of the underlying process; θ_k^* that varies with time τ_k will substitute for the fixed

θ^*. Naturally, $f(\cdot)$ is replaced by $f_k(\cdot)$. Specific examples will be discussed in Section 3.4 to appear.

To track the time variability of the sequence $\{\theta_k^*\}$, it is advisable that a_k in (2.1) should be a constant or is non-decaying (i.e., strictly bounded away from zero). It is well recognized that (2.1) with constant gain $a_k = a$ generally can track slight time variation and is of practical usage, see [6, Chap. 4].

Given that the optimizer θ_k^* is drifting over time, the classical SA theories for algorithms with decaying gains presented in Section 2.1 and relevant asymptotic properties (convergence and normality) are not directly transferable to the time-varying setup. Other notions of "convergence"—"concentration" to be more accurate—are developed for SA algorithms with non-decaying gain, especially those with constant gain, as summarized below.

2.2.1 Assumptions on Time-Varying Target

This section presents many assumed forms of the nonstationary drift for the underlying optimal values of the parameters. Prior works have considered time-varying problems under the R-M setting, with some hypothetical or empirical evolution forms for the underlying optimal values of the parameters θ_k^*. For example, [26, 71, 72] analyze a class of recursive algorithms after imposing the random-walk assumption on $\{\theta_k^*\}$. However, the random-walk model is suboptimal because the variance of the parameter sequence will explode to infinity over time. Besides, [29, Sect. 3.4] and many others assume that the

parameter can be estimated by KF, necessitating an explicit representation (e.g., linear state equation) for the evolution in $\{\theta_k^*\}$. Some general forms of the error bound for nonlinear and linear problems are discussed in [3, 76], still, on the basis that the knowledge-based description (a.k.a. state equation) for $\{\theta_k^*\}$ is available. Ref. [64] considers the limit as the rate of change of the functions goes to zero, yet it requires a Bayesian model for the changes in $\{f_k(\cdot)\}$. Admittedly, the tracking error characterization and the inference on the resulting estimates in the aforementioned works hinge upon the model assumptions and corresponding adaptive algorithms. To the best of our knowledge, there are no existing approaches in estimation theory that solve a sequence of time-varying problems, under only Assumption A.4 (to appear in Chapter 3) or B.4 (to appear in Chapter 4) without any further stringent state evolution assumption. Note that both A.4 and B.4 allow sporadic jumps in the sequence $\{\theta_k^*\}$. The relation between existing time-varying assumptions and ours is explained in Subsection 3.2.7.

Things become more complicated as the traditional continuous dynamics via a differential equation and discrete switching via jump process are not sufficient in modeling complex systems in finance [77], physics [46], or computer vision[40]. Hybrid diffusions can be modeled by a two-component Markov process, a continuous component (diffusion), and a discrete component (jump component). The change detection strategy discussed in Section 5.1 mainly focuses on detecting the jump component.

2.2.2 Criteria for Tracking Performance

This section lists a few metrics to evaluate the tracking performance of SA algorithms with non-decaying gains, especially with constant-gains. With a perpetually varying target θ_k^* and the inherent observation noise in either $y(\cdot) = f(\cdot) + \varepsilon(\cdot)$ or $\boldsymbol{Y}(\cdot) = \boldsymbol{g}(\cdot) + \xi(\cdot)$, there is no convergence per se. The concentration argument, that $\hat{\theta}_k$ stays within a neighborhood of θ_k^* in a certain statistical sense at time τ_k, is widely used in practical implementation. Often, we characterize the distance between $\hat{\theta}_k$ and θ_k^* using the MSE criteria $\mathbb{E}\|\hat{\theta}_k - \theta_k^*\|^2$. Note that the MSE is a family of criteria indexed by k. Under the mean-squared-error (MSE) tracking criteria, [6, Chap. 4] analyzes the tracking capability of recursive algorithms with a constant gain by assuming additional information on the state equation is available. References [34, 74] focus on the LMS algorithm in tracking time-varying solutions and presents an asymptotic stochastic big-O bound of the tracking error; however, the asymptotic bound is valid for linear models only and is not computable in general because of the higher than the fourth moments of the design vector required in the bound. There are also some finite-iteration error bounds developed under fairly strong assumptions. For an asymptotically stabilized target, i.e., $\lim_{k\to\infty} \theta_k^* = \theta^*$, [114] studied the quantification of the MSE bound. Chapter 3 to appear considers the problem of estimating unknown parameters and computing error bounds in a dynamic model. The finite-sample analysis of MSE for the estimates generated from (2.1) will be the central topic there. For finite-sample analysis, we list the distinction between [117] and our work in Subsection 3.1.3.

A form of convergence is possible for constant gains, typically based on limiting arguments as the gain magnitude gets small. The notable work [61, Chaps. 2–3], [65, Chaps. 7–10], [62, 63, 65] extensively illustrate the weak convergence method and its application in the constant-gain algorithms. Ref. [63] relates the limiting behavior of SA iterates in time-varying parameter identification problem to the asymptotic behavior of the limiting autonomous ODE, and thereupon establishes the theoretical foundation for the constant gain in accommodating general "time-varying parameter" identification problems: the estimates generated by the constant-gain SA algorithms tend to the true time-varying parameters, when both the constant gain and the time difference between two discrete sample points tend to zero. Later, [83] states similar results for constant-gain SA algorithm applied in constrained optimization. Nonetheless, such analysis only applies to the asymptotic behavior of SA estimates. In reality, neither the constant gain nor the time difference can go to zero in practice: a constant gain bounded away from zero is required, and the number of iterations per unit of time has to be finite. The weakly convergence limit for estimates generated from (2.6) will be discussed in Chapter 4, and a computable probabilistic bound will be provided under certain conditions. For weak-convergence argument, we list the distinction between other prior works and our work in Subsection 4.2.4.

For the fixed target case, i.e., $\theta_k^* = \theta^*$ for all k, one still cannot recover convergence a.s. of SA algorithms with a constant gain $a_k = a$ because the noise input is "persistent" as opposed to "asymptotically negligible" in the diminishing gain case. However, we are

able to say something about the limiting stationary distribution of $\hat{\theta}_k$, which is desirably centered near θ^*. Ref. [63] uses an ODE to approximate the asymptotic trajectory of the constant-gain SA iterates $\hat{\theta}_k$, which lays the foundation for the constant gain in accommodating time variability. It is proven that the estimate $\hat{\theta}_k$ generated by the constant-gain SA algorithms tends to the true time-varying parameter θ^*, when both the constant gain a and the time difference between two discrete sample points tend to zero. However, the asymptotic theory does not provide a practical gain-selection schema except for a vague expression "small a," let alone the stringent assumption of $\theta_k^* = \theta^*$ for all k. In addition to the discussion on constant-gain algorithms, [25] lends insight into non-diminishing gain selection to balance the bias-variance trade-off in the context of a stationary optimizer θ^*. References [79, 120] discuss some gain adaptation for stationary problems, as the "asymptotically optimal" stepsize can perform poorly in the practice [106, Sect. 4.5.3]. In general, a larger value of constant gain a helps the resulting iterates converging more quickly to the vicinity of the optimal parameter sequence $\{\theta_k^*\}$, corresponding to the state and measurement models; yet a smaller value of a increases the tracking stability. For asymptotically fixed targets such that $\lim_{k\to\infty} \theta_k^* = \theta^*$, [114] also analyzes the MSE decomposition for θ_k^*. Though both [25] and [114] consider time-varying objective functions $\{f_k\}$, the fixed or asymptotically fixed θ^* assumption limits their application in reality.

Other than the MSE criteria and the weak convergence argument, there are other streams in quantifying the tracking performance of constant-gain SA algorithms. Ref. [83]

analyzes the convergence properties in the small stepsize limit and the associated functional central limit theorem for fluctuations around the deterministic ODE limit. Ref. [52] establishes a law of iterated logarithms. The functional central limit theorem characterizing a Gauss-Markov process as a limit in law of suitably scaled fluctuations is also used for suggesting performance metrics for tracking application, see [7].

2.3 Supporting Materials in ODE

One useful method to analyze the property of SA estimates is to relate the iterates $\hat{\theta}_k$ to the trajectory of an initial value problem (IVP). The ODE in this IVP is determined by the average dynamics of the algorithm. Chapter 4 will discuss the ODEs defined by the dynamics projected onto a compact constraint set denoted as Θ. The solutions to such ODEs will be the weakly convergence limits of the paths of constrained SA algorithms. A basic constrained or projected SA algorithm is

$$\hat{\theta}_{k+1} = \mathscr{P}_{\Theta}\big(\hat{\theta}_k - a_k \hat{\boldsymbol{g}}_k(\hat{\theta}_k)\big),\ k \in \mathbb{N}, \tag{2.6}$$

where $\Theta \subsetneq \mathbb{R}^p$ is closed and bounded, $\mathscr{P}_{\Theta}(\zeta) = \operatorname{argmin}_{\theta \in \Theta} \|\theta - \zeta\|$, and $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ can also be decomposed as (2.2). This section reviews some supporting materials for ODEs that facilitate the analysis of (2.6).

2.3.1 Limits of Sequence of Continuous Functions

The extended Arzelà-Ascoli Theorem reviewed in this subsection will be useful in extracting weak-convergent subsequence whose limits satisfy the mean ODE in Section 4.1. Let $C^j\left(\mathbb{R} \mapsto \mathbb{R}^p\right)$ be the space of functions that map from $\mathbb{R}$ to $\mathbb{R}^p$ and are jth-order continuous. Usually, $C^0(\mathbb{R} \mapsto \mathbb{R}^p)$ is compactly written as $C(\mathbb{R} \mapsto \mathbb{R}^p)$. We can similarly define $C^j\left([l, r] \mapsto \mathbb{R}^p\right)$, $C^j\left([0, \infty) \mapsto \mathbb{R}^p\right)$, where l and r are real numbers. The metric for both $C^j\left([l, r] \mapsto \mathbb{R}^p\right)$ is the supremum norm, and the metric for $C^j\left(\mathbb{R} \mapsto \mathbb{R}^p\right)$ and $C^j\left([0, \infty) \mapsto \mathbb{R}^p\right)$ is the local supremum norm. For example, a sequence of functions $\{\boldsymbol{f}_k(\cdot)\}$ in $C(\mathbb{R} \mapsto \mathbb{R}^p)$ converges to zero if it converges to zero uniformly on every bounded time interval within the domain of definition.

Definition 2.3.1 (Equicontinuous)**.** Let the function sequence $\{\boldsymbol{f}_k(\cdot)\}$ indexed by k be a subset of $C\left(\mathbb{R} \mapsto \mathbb{R}^p\right)$. The function sequence $\{\boldsymbol{f}_k(\cdot)\}$ is said to be equicontinuous if (1) $\{\boldsymbol{f}_k(0)\}$ is bounded for all k; and (2) for each $T > 0$ and $\varepsilon > 0$, there exists a $\delta > 0$ such that $\sup_{0 \leq t-s \leq \delta, |t| \leq T} \|\boldsymbol{f}_k(t) - \boldsymbol{f}_k(s)\| \leq \varepsilon$ for all k.

Theorem 2.3.1 (Arzelà-Ascoli)**.** *If the function sequence $\{\boldsymbol{f}_k(\cdot)\}$ is equicontinuous in the function space $C\left(\mathbb{R} \mapsto \mathbb{R}^p\right)$, then there exists a subsequence that converges to some function in $C\left(\mathbb{R} \mapsto \mathbb{R}^p\right)$, uniformly on each bounded interval.*

Definition 2.3.2 (Equicontinuous in the extended sense)**.** Let $\boldsymbol{f}_k(\cdot) : \mathbb{R} \mapsto \mathbb{R}^p$ be measurable for every k. Note that $\boldsymbol{f}_k(\cdot)$ is not necessarily continuous. The function sequence $\{\boldsymbol{f}_k(\cdot)\}$ is said to be equicontinuous in the extended sense if (1) $\{\boldsymbol{f}_k(0)\}$ is

bounded for all k, and (2) for each $T > 0$ and $\varepsilon > 0$, there exists a $\delta > 0$ such that $\limsup_k \sup_{0 \le t-s \le \delta, |t| \le T} \| \boldsymbol{f}_k(t) - \boldsymbol{f}_k(s) \| \le \varepsilon$.

Theorem 2.3.2 (Extended Arzelà-Ascoli)**.** *If the function sequence* $\{\boldsymbol{f}_k(\cdot)\}$ *is equicontinuous in the extended sense, then there exists a subsequence that converges to a function in* $C(\mathbb{R} \mapsto \mathbb{R}^p)$*, uniformly on each bounded interval.*

2.3.2 Existence and Uniqueness of the Result

The regularity conditions to ensure the existence and uniqueness of the solution to an IVP reviewed in this subsection will be applied on the average ODE in Chapter 4.

Definition 2.3.3 (Locally Lipschitz continuous)**.** Let $\boldsymbol{f}$ be a $\mathbb{R}^p$-valued function that takes input arguments (t, θ) within the open domain $U \subseteq \mathbb{R}^{p+1}$. $\boldsymbol{f}$ is said to be locally Lipschitz continuous in θ uniformly w.r.t. t, if

$$\sup_{(t,\theta_1) \neq (t,\theta_2) \in V} \frac{\| \boldsymbol{f}(t, \theta_1) - \boldsymbol{f}(t, \theta_2) \|}{\| \theta_1 - \theta_2 \|} < \infty, \tag{2.7}$$

for every compact subset $V \subsetneq U$.

Definition 2.3.4. We say that $\boldsymbol{f} \in C^0_{\text{Lips}}(U \mapsto \mathbb{R}^p)$ with $U \subseteq \mathbb{R}^{p+1}$ being open, if $\boldsymbol{f}$ is zeroth-order continuous in θ, and is locally Lipschitz continuous in θ uniformly w.r.t. t. Usually, $C^0_{\text{Lips}}(U \mapsto \mathbb{R}^p)$ will be compactly written as $C_{\text{Lips}}(U \mapsto \mathbb{R}^p)$.

Remark 1. Note that Definition 2.3.4 does not convey any information regarding whether $\boldsymbol{f}$ is continuous in t.

Consider the following IVP:

$$\begin{cases} \dfrac{d}{dt}\boldsymbol{\theta}(t) = \boldsymbol{f}\left(t, \boldsymbol{\theta}\right), \ t \geq t_0, \\ \boldsymbol{\theta}\left(t_0\right) = \hat{\boldsymbol{\theta}}_0 \, . \end{cases} \tag{2.8}$$

When dealing with IVP (2.8), we often suppose that $\boldsymbol{f} \in C_{\text{Lips}}(U \mapsto \mathbb{R}^p)$ and the domain $U \subseteq \mathbb{R}^{p+1}$ is open.

Theorem 2.3.3 (Picard-Lindelföf)**.** *[110, Thm. 2.2] Suppose $\boldsymbol{f} \in C_{\text{Lips}}(U \mapsto \mathbb{R}^p)$, where $U \subseteq \mathbb{R}^{p+1}$ is open. Then there exists a unique local solution $\boldsymbol{Z}\left(t\right) \in C^1\left(I \mapsto \mathbb{R}^p\right)$ of the IVP (2.8), where $I \subsetneq \mathbb{R}$ is some interval around t_0.*

For example, let M be the maximum of $\|f\|$ on $[t_0, t_0 + T] \times \overline{\text{Ball}_\delta(\hat{\boldsymbol{\theta}}_0)} \subsetneq U$. The solution $\boldsymbol{Z}(t)$ exists at least for $t \in [t_0, t_0 + T_0]$ and remains within $\overline{\text{Ball}_\delta(\hat{\boldsymbol{\theta}}_0)}$, where $T_0 = \min\{T, \delta/M\}$. The analogous result holds for $[t_0 - T_0, t_0]$.

Corollary 2.3.1. *[110, Lem. 2.3] Suppose $\boldsymbol{f} \in C^j(U \mapsto \mathbb{R}^p)$ for $j \geq 1$ where $U \subseteq \mathbb{R}^{p+1}$ is open and $(t_0, \hat{\boldsymbol{\theta}}_0) \in U$. Then there exists a unique local solution $\boldsymbol{Z}(t) \in C^{j+1}(I \mapsto \mathbb{R}^p)$ of the IVP (2.8), where $I \subsetneq \mathbb{R}$ is some interval around t_0.*

Theorem 2.3.4 (Improved Picard-Lindelföf)**.** *[110, Thm. 2.5] Suppose $\boldsymbol{f} \in C_{\text{Lips}}(U \mapsto \mathbb{R}^p)$ where $U \subseteq \mathbb{R}^{p+1}$ is open. Choose $(t_0, \hat{\boldsymbol{\theta}}_0) \in U$ and $\delta, T > 0$ such that $[t_0, t_0 + T] \times$*

$\overline{\text{Ball}_\delta(\hat{\boldsymbol{\theta}}_0)} \subsetneq U$. *Set*

$$\begin{cases} M(t) = \displaystyle\int_{t_0}^{t} \sup_{\boldsymbol{\theta}\in\text{Ball}_\delta(\hat{\boldsymbol{\theta}}_0)} \|\boldsymbol{f}(s,\boldsymbol{\theta})\| ds \\ \mathscr{L}(t) = \displaystyle\sup_{\boldsymbol{\theta}_1\neq\boldsymbol{\theta}_2\in\text{Ball}_\delta(\hat{\boldsymbol{\theta}}_0)} \frac{\|\boldsymbol{f}(t,\boldsymbol{\theta}_1) - \boldsymbol{f}(t,\boldsymbol{\theta}_2)\|}{\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|}. \end{cases} \tag{2.9}$$

Define T_0 *as* $T_0 \equiv \sup\{0 < t \leq T | M(t_0 + t) \leq \delta\}$, *which is well-defined because* $M(t)$ *is nondecreasing in* t. *Suppose* $L_1(T_0) \equiv \int_{t_0}^{t_0+T_0} \mathscr{L}(s)\, ds < \infty$. *Then there exists a unique local solution* $\boldsymbol{Z}(t) \equiv \lim_{m\to\infty}[\boldsymbol{K}^m(\hat{\boldsymbol{\theta}}_0)](t) \in C^1\left([t_0, t_0+T_0] \mapsto \overline{\text{Ball}_\delta(\hat{\boldsymbol{\theta}}_0)}\right)$ *of the IVP (2.8), where* $[\boldsymbol{K}(\boldsymbol{\theta})](t) = \hat{\boldsymbol{\theta}}_0 + \int_{t_0}^{t} \boldsymbol{f}(\boldsymbol{\theta}(s), s)\, ds$, *and satisfies* $\sup_{t_0\leq t\leq t_0+T_0} \|\boldsymbol{Z}(t) - \boldsymbol{K}^m(\hat{\boldsymbol{\theta}}_0)(t)\| \leq \frac{L_1(T_0)^m}{m!} e^{L_1(T_0)} \int_{t_0}^{t_0+T_0} \|\boldsymbol{f}(s,\hat{\boldsymbol{\theta}}_0)\| ds$.

In fact, the continuity of $\boldsymbol{f}$ is not necessary to ensure the existence of a local solution $\boldsymbol{Z}(t)$. For $\boldsymbol{Z}(t)$ to exist locally, all we need are (i) $\boldsymbol{f}$ is measurable, (ii) $M(t)$ is finite, and (iii) $\mathscr{L}(t)$ is locally integrable in terms of $\int_I \mathscr{L}(s)\, ds < \infty$ for any compact interval I. However, under less stringent conditions, the solution $\boldsymbol{Z}(\cdot)$ may no longer fall in $C^1(I \mapsto \mathbb{R}^p)$.

Corollary 2.3.2 (Extension Theorem)**.** *[110, Corr. 2.6] Suppose* $[t_0, T] \times \mathbb{R}^p \subsetneq U$ *and* $\int_{t_0}^{T} \mathscr{L}(s)\, ds < \infty$ *where* $\mathscr{L}(t)$ *is defined in (2.9), then* $\boldsymbol{Z}(t)$ *is well-defined for all* $t \in [t_0, T]$. *In particular, if* $U = \mathbb{R}^{p+1}$ *and* $\int_{-T}^{T} \mathscr{L}(s)\, ds < \infty$ *for* all $T > 0$, *then* $\boldsymbol{Z}(t)$ *is well-defined for all* $t \in \mathbb{R}$.

In real-world applications, the iterates are usually confined within a compact set $\boldsymbol{\Theta}$ as in (2.6). If an iterate ever leaves $\boldsymbol{\Theta}$, it is immediately sent back to the closest point in $\boldsymbol{\Theta}$. In

accordance with the constrained SA algorithm (2.6), we are interested in

$$\begin{cases} \dot{\boldsymbol{\theta}}(t) = \boldsymbol{f}(t, \boldsymbol{\theta}) + \boldsymbol{h}(t), \quad \boldsymbol{h}(t) \in -\mathrm{Cone}(\boldsymbol{\theta}(t)), \\ \boldsymbol{\theta}(0) = \hat{\boldsymbol{\theta}}_0, \end{cases} \tag{2.10}$$

where $\boldsymbol{h}(\cdot)$ is the minimum force needed to keep $\boldsymbol{\theta}(\cdot)$ within $\boldsymbol{\Theta}$. Specifically, for $\boldsymbol{\theta} \in \mathrm{int}(\boldsymbol{\Theta})$, $\mathrm{Cone}(\boldsymbol{\theta}(t))$ contains $\mathbf{0}$ only; for $\boldsymbol{\theta} \in \partial\boldsymbol{\Theta}$, $\mathrm{Cone}(\boldsymbol{\theta}(t))$ is the convex cone generated by the set of outward normals at $\boldsymbol{\theta}(t)$ of the faces on which $\boldsymbol{\theta}(t)$ lies, and therefore $\boldsymbol{h}(t)$ points inward.

2.3.3 Alekseev's Formula

The Alekseev's formula reviewed in this subsection will facilitate deriving a computable probabilistic bound in Section 4.2.

Definition 2.3.5 (Fundamental Matrix)**.** Let $\boldsymbol{A}(t) \in \mathbb{R}^{p\times p}$ for all t. A fundamental matrix of a system of p homogeneous ODEs $\dot{\boldsymbol{z}}(t) = \boldsymbol{A}(t)\boldsymbol{z}(t)$ is a matrix-valued function $\boldsymbol{\Phi}(t)$ whose columns are linearly independent solutions of the ODE.

A useful tool for bounding the errors resulted from tolerable perturbations is the Alekseev's formula [1]. Consider the IVP (2.8) and its perturbed system

$$\begin{cases} \dfrac{d}{dt}\boldsymbol{\zeta}(t) = \boldsymbol{f}(t, \boldsymbol{\zeta}) + \boldsymbol{e}(t, \boldsymbol{\zeta}), \ t \geq t_0, \\ \boldsymbol{\zeta}(t_0) = \hat{\boldsymbol{\theta}}_0. \end{cases} \tag{2.11}$$

Assume that $\boldsymbol{f} : \mathbb{R} \times \mathbb{R}^p \mapsto \mathbb{R}^p$ appearing in both (2.8) and (2.11) is measurable in t and continuously differentiable in θ with bounded derivatives uniformly w.r.t. t. Further assume that $\boldsymbol{e} : \mathbb{R} \times \mathbb{R}^p \mapsto \mathbb{R}^p$ appearing in (2.11) is measurable in t and Lipschitz in θ uniformly w.r.t. t.

Let $\theta(t; t_0, \hat{\theta}_0)$ and $\zeta(t; t_0, \hat{\theta}_0)$ denote respectively the unique solutions to (2.8) and (2.11) for $t \geq t_0$ with initial condition $\theta(t_0; t_0, \hat{\theta}_0) = \zeta(t_0; t_0, \hat{\theta}_0) = \hat{\theta}_0$. Then for $t \geq t_0$, we have the following representation:

$$\zeta(t; t_0, \hat{\theta}_0) = \theta(t; t_0, \hat{\theta}_0) - \int_{t_0}^{t} \left[\Phi\big(t; s, \zeta(s; t_0, \hat{\theta}_0)\big)\, \boldsymbol{e}\big(s, \zeta(s; t_0, \hat{\theta}_0)\big) \right] ds, \tag{2.12}$$

where $\Phi(t; s, \hat{\theta}_0)$ for any $\hat{\theta}_0 \in \mathbb{R}^p$ is the fundamental matrix of the linearized system

$$\dot{\boldsymbol{z}}(t) = \left. \frac{\partial \boldsymbol{f}(t, \theta)}{\partial \theta^T} \right|_{\theta = \theta(t; s, \hat{\theta}_0)} \cdot \boldsymbol{z}(t), \quad \text{for } t \geq s, \tag{2.13}$$

such that $\Phi(s; s, \hat{\theta}_0) = \boldsymbol{I}_p$.

2.3.4 Stability for Nonautonomous System

The notion of stability for nonautonomous system reviewed in this subsection will be applied on the average ODE in Section 4.1.

For IVP (2.8), take $t_0 = 0$ w.l.o.g.

Definition 2.3.6 (Equilibrium of Unconstrained Nonautonomous System)**.** The equilibrium point θ^* of the IVP (2.8) is such that $\boldsymbol{f}(t, \theta^*) = \boldsymbol{0}$ for all $t \geq t^*$ with $t^* \geq 0$.

When an equilibrium exists, the system state remains at θ^* once it reaches θ^*. By a suitable transformation, we can make the equilibrium point of the transformed system to be the origin $\mathbf{0}$. With abuse of notation, we use (2.8) to represent the transformed system whose equilibrium point is at the origin within the rest of this subsection.

Definition 2.3.7 (Stable)**.** The IVP (2.8), whose equilibrium is the origin $\mathbf{0}$, is said to be stable at t^* if, for any $\varepsilon > 0$, there exists a real number $\delta = \delta(\varepsilon, t^*) > 0$ such that $\|\theta(0)\| \leq \delta$ implies $\|\theta(t)\| \leq \varepsilon$ for all $t \geq t^*$.

Definition 2.3.8 (Convergent)**.** The IVP (2.8), whose equilibrium is the origin $\mathbf{0}$, is said to be convergent at t^* if, there exists a real number $\delta = \delta(t^*) > 0$ such that $\|\theta(0)\| \leq \delta$ implies $\lim_{t\to\infty} \theta(t) = \mathbf{0}$.

Definition 2.3.9 (Asymptotically Stable)**.** The IVP (2.8), whose equilibrium is the origin $\mathbf{0}$, is said to be asymptotically stable at time t^* if it is both stable and convergent at t^*.

Definitions 2.3.7–2.3.9 can be strengthened to "uniformly stable," "uniformly convergent," and "uniformly asymptotically stable" respectively, if the dependence on t^* can be removed from the defining statements. In fact, the uniformity in time is important to ensure the attraction region does not vanish as time varies.

Definition 2.3.10 (Lyapunov's Stability)**.** A set $A \subseteq \mathbb{R}^p$ is said to be locally stable in the sense of Liapunov if, for each $\varepsilon > 0$, there exists a $\delta > 0$ such that all trajectories starting from $\hat{\theta}_0 \in \mathrm{Ball}_\delta(A)$ will never leave $\mathrm{Ball}_\varepsilon(A)$. If the trajectories ultimately go to A, then

A is said to be asymptotically stable in the sense of Liapunov. If this holds for all initial conditions, then the asymptotic stability is said to be global.

2.4 Review on Weak Convergence

This section lays out basic facts in weak convergence theory. "Weak convergence" of the function-valued random variables (a.k.a. random function) extends the notion of "convergence in distribution" of the $\mathbb{R}^p$-valued random variables. Let $D(\mathbb{R} \mapsto \mathbb{R}^p)$ be the space of functions that map from $\mathbb{R}$ to $\mathbb{R}^p$ and are right-continuous with left-hand limits. We are interested in the function space $D(\mathbb{R} \mapsto \mathbb{R}^p)$ equipped with the Skorohod topology [13, Sect. 12]. The exact definition of the Skorohod topology is somewhat technical[2] and not essential for the upcoming proofs. Under the Skorohod topology, $D(\mathbb{R} \mapsto \mathbb{R}^p)$ is separable and metrizable, and the metric is complete.

The function space $D(\mathbb{R} \mapsto \mathbb{R}^p)$ is useful for two reasons. First of all, the processes with paths in $D(\mathbb{R} \mapsto \mathbb{R}^p)$ come up naturally in applications. Moreover, the Skorohod topology in $D(\mathbb{R} \mapsto \mathbb{R}^p)$ is an extension of the topology of uniform convergence on bounded time intervals in $C(\mathbb{R} \mapsto \mathbb{R}^p)$, in that a local (k, t)-dependent stretching or contraction of the time scale is allowed where k is the index of the function sequence and t lies within the domain of the kth function. Therefore, this topology is weaker than that of $C(\mathbb{R} \mapsto \mathbb{R}^p)$,

[2]For a function $\lambda(\cdot)$ in the space Λ_T of strictly increasing, continuous mappings of $[0, T]$ onto itself, first define $\|\lambda\| = \sup_{s<t<T} |\log[(\lambda(t) - \lambda(s))/(t - s)]|$. The distance between $x(\cdot)$ and $y(\cdot)$ in $D([0, T] \mapsto [0, T])$ is given by $\inf_{\lambda \in \Lambda_T}\{\|\lambda\| + \sup_{t \in [0,T]} |x(t) - y(\lambda(t))|\}$. The Skorohod topology on the space $D([0, \infty) \mapsto [0, \infty))$ or $D(\mathbb{R} \mapsto \mathbb{R})$ can be defined by requiring the convergence in the Skorohod metric on each compact interval $[0, T]$ or $[-T, T]$ for $T > 0$. The metric on the product space $D(\mathbb{R} \mapsto \mathbb{R}^p)$ can be taken to be the sum of the metrics on the component spaces.

so that the criteria for compactness are less stringent, even if the paths or their limits may still lie within $C(\mathbb{R} \mapsto \mathbb{R}^p)$. Consequently, this property is useful in dealing with "nice" discontinuities such that the discontinuities do not appear in the limit. Second, the convergence in such space has many important ramifications (see further details [60, Chap. 2]). What matters the most to us is that the convergence of a sequence of functions in $D(\mathbb{R} \mapsto \mathbb{R}^p)$ to a continuous function in $C(\mathbb{R} \mapsto \mathbb{R}^p)$ in the Skorohod topology is equivalent to convergence uniformly on each bounded time interval in $C(\mathbb{R} \mapsto \mathbb{R}^p)$.

2.4.1 Weak Convergence

This section reviews the notion of weak convergence of random function by making an analogy with the weak convergence of random variable, and they will be used in the proofs of Subsection 4.1.4. Let us work on a common probability space $(\Omega, \mathbb{P}, \mathcal{A})$.

Definition 2.4.1 (Weak Convergence of Random Variables)**.** [13] A sequence of $\mathbb{R}^p$-valued random variables $\boldsymbol{X}_k(\omega)$ indexed by k is said to converge in distribution to a $\mathbb{R}^p$-valued random variable $\boldsymbol{X}(\omega)$ if and only if $\mathbb{E}F\left(\boldsymbol{X}_k(\omega)\right) \to \mathbb{E}F\left(\boldsymbol{X}(\omega)\right)$ as $k \to \infty$ for every bounded and continuous function $F : \mathbb{R}^p \mapsto \mathbb{R}$.

The weak convergence in Definition 2.4.1 is also known as "convergence in distribution" and can be compactly written as $\boldsymbol{X}_k \xrightarrow{\text{dist}} \boldsymbol{X}$ by suppressing the ω dependence. To distinguish random variables and random processes, we represent random processes using $\boldsymbol{x}(\cdot, \cdot)$ and $\boldsymbol{x}$ takes $t \in \mathbb{R}$ and $\omega \in \Omega$ as inputs. For each fixed time t, $\boldsymbol{x}(t, \cdot)$ is a random variable. For each fixed sample ω, $\boldsymbol{x}(\cdot, \omega)$ is a function of time. We say

that $\boldsymbol{x}(t, \omega)$ is measurable if $\boldsymbol{x}(t, \omega) : \mathbb{R} \times \Omega \mapsto \mathbb{R}^p$ is $\mathcal{B}(\mathbb{R}) \times \mathcal{A}$-measurable, where $\mathcal{B}(\mathbb{R})$ is the Borel sigma-field on the real line. We say that a random process $\boldsymbol{x}(t, \omega)$ is (almost surely) continuous if, for (almost) every $\omega \in \Omega$, $\boldsymbol{x}(\cdot, \omega)$ is continuous w.r.t. t. Similar to writing the random variable $\boldsymbol{X}(\omega)$ as $\boldsymbol{X}$, we write $\boldsymbol{x}(\cdot, \cdot)$ compactly as $\boldsymbol{x}(\cdot)$ by suppressing the ω dependence; i.e., $\boldsymbol{x}(\cdot)$ is a random function (of time t). The following definition of the weak convergence of random functions is a natural extension of Definition 2.4.1.

Definition 2.4.2 (Weak Convergence of Random Functions)**.** [88] The weak convergence (a.k.a. convergence in distribution) $\boldsymbol{x}_k(\cdot) \xrightarrow{\text{dist}} \boldsymbol{x}(\cdot)$ in $D(\mathbb{R} \mapsto \mathbb{R}^p)$ is equivalent to $\mathbb{E}\boldsymbol{F}(\boldsymbol{x}_k(\cdot, \omega)) \to \mathbb{E}\boldsymbol{F}(\boldsymbol{x}(\cdot, \omega))$ as $k \to \infty$ for any bounded and continuous function $\boldsymbol{F}$ that maps $\boldsymbol{x}(\cdot) \in D(\mathbb{R} \mapsto \mathbb{R}^p)$ to $\left(\boldsymbol{x}(t_1)^T, \cdots, \boldsymbol{x}(t_l)^T\right)^T \in \mathbb{R}^{pl}$ for $t_1, \cdots, t_l \in \mathbb{R}$ and $l \geq 1$.

2.4.2 Tightness

This section reviews the notion of tightness and a set of sufficient conditions to prove tightness, which will be useful in Subsection 4.1.4.

The sequence of random variables $\{\boldsymbol{X}_k\}$ is said to be tight (a.k.a. uniformly bounded in probability) if, for each $\delta \in (0, 1]$, there exists a compact set $\mathcal{S}_\delta$ such that $\mathbb{P}\{\boldsymbol{X}_k \in \mathcal{S}_\delta\} \geq 1 - \delta$ for all k. The Helly–Bray theorem states that a tight sequence must have a further subsequence that converges weakly. We also define the notion of sequential compactness by "each subsequence contains a further subsequence that converges weakly."

The definition of tightness carries over to random functions $\boldsymbol{x}_k(\cdot)$. The exact statement of the tightness of random functions is technical and we will mainly use the following supporting lemma to facilitate proving tightness of a sequence of random functions $\boldsymbol{x}_k(\cdot)$ within $D(\mathbb{R} \mapsto \mathbb{R}^p)$.

Lemma 2.4.1. *Let $\theta_k(\cdot)$ be a sequence of random processes indexed by k with paths in $D(\mathbb{R} \mapsto \mathbb{R}^p)$. If the following two conditions hold, then we claim that $\{\theta_k(\cdot)\}$ is tight in $D(\mathbb{R} \mapsto \mathbb{R}^p)$.*

1. *Compact containment condition. For each $\delta \in (0, 1]$, and for each t in a dense subset of $\mathbb{R}$, there exists a compact set $\mathcal{S}_{\delta,t} \subsetneq \mathbb{R}^p$ such that*

$$\inf_k \mathbb{P}\{\theta_k(t) \in \mathcal{S}_{\delta,t}\} \geq 1 - \delta. \tag{2.14}$$

2. *For each $T > 0$,*

$$\lim_{\tau \to 0} \limsup_{k \to \infty} \sup_{0 \leq s \leq \tau, |t| \leq T} \mathbb{E}\|\theta_k(t+s) - \theta_k(t)\| = 0. \tag{2.15}$$

Remark 2. (2.15) does not imply the continuity of the paths of either $\theta_k(\cdot)$ or any weak sense limit $\theta(\cdot)$.

Proof of Lemma 2.4.1. Under the metric defined in [13, Sect. 12], the metric space $D(\mathbb{R} \mapsto \mathbb{R}^p)$ is separable and complete. Prohorov's theorem [33, p. 104] states that

tightness is equivalent to sequential compactness on a complete[3] separable[4] metric space. By Prohorov's Theorem, any sequence in $D(\mathbb{R} \mapsto \mathbb{R}^p)$ is tight if and only if it is relatively compact. The result follows from [60, Thm. 2.7 on p. 10]. □

If a sequence $\{\boldsymbol{x}_k(\cdot)\}$ on a complete separable metric space (mainly $D(\mathbb{R} \mapsto \mathbb{R}^p)$ in our discussion) is shown to be tight through Lemma 2.4.1 then it must have a weakly convergent subsequence. The proposition that "if a sequence of random functions is tight, then it has a weak convergent subsequence" is in fact an extension of the proposition that "if a sequence of random variables is tight, then it has a subsequence that converges in distribution to some random variable."

Lemma 2.4.2. *[65, p. 230] Suppose that a sequence of processes* $\{\Theta_k(\cdot)\}$ *is tight in* $D(\mathbb{R} \mapsto \mathbb{R}^p)$ *and that on each interval* $[-T, T]$ *the size of the maximum discontinuity goes to zero in probability as* $k \to \infty$*, then any weak sense limit process must have continuous paths w.p.1.*

2.4.3 Skorohod Embedding

This section reviews the notion of Skorohod embedding, which will be applied in the proofs of Subsection 4.1.4. Recall that $D(\mathbb{R} \mapsto \mathbb{R}^p)$ is a complete and separable metric space with the metric $d(\cdot, \cdot)$ that metricizes the Skorohod topology.

Theorem 2.4.1 (Skorohod representation)**.** *Let* $\Theta_k(\cdot) \xrightarrow{\text{dist}} \Theta(\cdot)$ *for* $\Theta_k(\cdot), \Theta(\cdot) \in D(\mathbb{R} \mapsto \mathbb{R}^p)$*. There exists a probability space* $(\tilde{\Omega}, \tilde{\mathcal{B}}, \tilde{\mathbb{P}})$ *with associated random functions* $\tilde{\Theta}_k(\cdot)$ *in*

[3] A metric space is complete if every Cauchy sequence in it converges to a point in it.
[4] A topological space is separable if it contains a countable dense subset.

$D(\mathbb{R} \mapsto \mathbb{R}^p)$ and $\tilde{\boldsymbol{\theta}}(\cdot)$ defined on it such that for each dense[5] *set $\mathcal{S} \subsetneq D(\mathbb{R} \mapsto \mathbb{R}^p)$:*

$$\tilde{\mathbb{P}}\{\tilde{\boldsymbol{\theta}}_k(\cdot) \in \mathcal{S}\} = \mathbb{P}\{\boldsymbol{\theta}_k(\cdot) \in \mathcal{S}\},\ \tilde{\mathbb{P}}\{\tilde{\boldsymbol{\theta}}(\cdot) \in \mathcal{S}\} = \mathbb{P}\{\boldsymbol{\theta}(\cdot) \in \mathcal{S}\}, \tag{2.16}$$

and $d(\tilde{\boldsymbol{\theta}}_k(\cdot), \tilde{\boldsymbol{\theta}}(\cdot)) \to 0$ w.p.1.

W.l.o.g., we suppose that the probability space is carefully chosen so that weak convergence is equivalent to convergence w.p.1 uniformly on bounded time intervals. Note that the use of the Skorohod representation itself does not imply that the original sequence $\boldsymbol{\theta}_k(\cdot)$ converges w.p.1.

[5] A subset B of a topological space A is dense if every point in A either belongs to B or a is a limit point of B.

Chapter 3

Tracking Capability

This chapter focuses on the analysis of the tracking performance of adaptive-gain SA algorithms. Time-varying optimization problems arise frequently, including in deterministic nonlinear programming [8], e.g., the method of penalty functions involves selecting a growing[1] sequence of the penalty coefficients and solve for the constrained minimization problem sequentially for each iteration. Aside from the underlying controllable parameter θ, the loss function $f(\cdot)$ may also depend on some other factors, such as time. Following the motivations discussed in Section 1.1, we are mainly interested in situations where the noisy information of the time-varying loss functions can be collected at sampling time τ_k corresponding to discrete index k. That is, this chapter considers a sequence of loss functions $f_k(\theta)$ at sampling times τ_k, instead of one single loss function $f(\cdot)$ that remains unchanged. Moreover, only a small number (either one or two) of noisy

[1] With this perspective, it is easier to study the behavior of a solution at infinity and also estimate the strategy of the choice of the sequence of penalty coefficients.

observations pertaining to $f_k(\cdot)$ are revealed at the sampling time τ_k. Such setup contrasts with [86, 97] in that they assume *noise-free* observations. It is also different from [117], where as many sequential measurements as needed can be collected at each discrete time instance. In this setting, we only require at most two parallel measurements. The meanings of "sequential" and "parallel" will be explained in Subsection 3.1.3.

This chapter is dedicated to showing the tracking capability of SA algorithms with non-decaying gain as applied in a time-varying framework, where a sequence of loss functions $\{f_k(\cdot)\}$ changes along $[\tau_0, \tau_K]$ of our interest with K being the last sampling index, and a slowly time-varying optimum. By "slowly" we mean that the average distance between successive optimizers is strictly bounded from above on average; infrequent jumps are allowed in such a setting. Section 3.1 presents the problem setup and Section 3.2 discusses the model assumptions. Section 3.3 establishes the tracking capability by computing the error bound for MAD and RMS. Section 3.4 discusses some special cases of (2.1) for nonlinear root-finding.

3.1 Problem Formulation

This section introduces necessary concepts arising in the parameter estimation and states the target-tracking problem.

3.1.1 Basic Setup of Time-Varying SA Problems

We consider the problem of estimating a time-varying parameter $\{\theta_k^*\}$ that varies "slowly" and formulate the problem from an online convex optimization perspective. Each f_k in the sequence of convex functions $\{f_k(\cdot)\}$ indexed by k is a differentiable mapping from $\mathbb{R}^p$ to $\mathbb{R}$. Recall that each index k corresponds to the actual time τ_k. Within this chapter, let us suppose that the sampling frequency is bounded from above; i.e., the actual time elapsed between two consecutive samples, $\tau_{k+1} - \tau_k$, is bounded from below by zero for all $k \in \mathbb{N}$. Note that the sampling intervals $\tau_{k+1} - \tau_k$ need *not* remain constant across k.

Our goal is to efficiently track the value(s) of θ that minimizes instantaneous scalar-valued loss function (sequence) $f_k(\cdot)$:

$$\text{Find } \theta_k^* \equiv \arg\min_{\theta \in \Theta} f_k(\theta) \text{ for each } k \in \mathbb{N}. \tag{3.1}$$

Let $\hat{\theta}_k$, whose recursive scheme will soon be discussed in the next subsection, represent our best possible estimate for parameter θ_k^* at time τ_k. The experimenter does not know the exact functional form of $f_k(\cdot)$, but can receive instant feedback immediately after the decision $\hat{\theta}_k$ is selected. Usually, the instant feedback regarding $f_k(\cdot)$ at a design point $\hat{\theta}_k$ is either a noisy realization of the cost or a noisy evaluation of the gradient information

$$\begin{cases} y_k(\theta) = f_k(\theta) + \varepsilon_k(\theta), & (3.2) \\ \boldsymbol{Y}_k(\theta) = \dfrac{\partial f_k(\theta)}{\partial \theta} + \xi_k(\theta). & (3.3) \end{cases}$$

A comprehensive summary of gradient estimation methods available through the mid-1990s is [102] and some recent detailed analysis of some of these methods is given in [14]. Note that (3.3) differs from (2.5) in that both of the terms on the r.h.s. of (3.3) vary with k whereas only $\xi_k(\cdot)$ on the r.h.s. of (2.5) depends on k.

Remark 3. We need to clarify both $f_k(\cdot)$ and its minimizer θ_k^* are *deterministic* to the experimenter. Granted, θ_k^* itself can evolve stochastically and a common example is that the state space model in KF involves a multivariate normal distribution. Nonetheless, the randomness in θ_k^* will *not* be taken into account while formulating the loss function at time τ_k, an example of which is (3.61) to appear. That said, at time τ_k, the loss function $f_k(\cdot)$ is formulated in a way that θ_k^* is deemed as a fixed value, and only the measurement noise $\varepsilon_k(\cdot)$ in (3.2) or $\xi_k(\cdot)$ (3.3) is taken into consideration.

The general setting (3.1)–(3.3) subsumes many target tracking scenarios where θ_k^* represents the locations of the targets being pursued by one or more agents. The agents are expected to utilize the immediate feedback via either (3.2) or (3.3) to improve their estimates $\hat{\theta}_k$ for parameter θ_k^* in an online fashion. Often, at each sampling time τ_k, only a *few* (either one or two in our discussion) noisy measurements, either in the form of (3.2) or (3.3), can be gathered, and the evaluation point is at the agents' disposal. In the defense applications, the agents only observe the target's location when necessary, because frequent emission of radar signals inevitably and undesirably reveals the agents' position. Such a setting promotes the "few measurements at each time" requirement.

3.1.2 SA Algorithm with Non-Decaying Gain

We are interested in characterizing SA algorithms [106, Eq. (6.5) on p. 157], namely, (2.1) with a non-diminishing step size. Note that a_k has to be strictly bounded away from zero, and advance tuning is required (see more details in Algorithm 1). Let us briefly discuss the several forms of $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ here, corresponding to the two feedback forms (3.2) and (3.3), respectively.

- When the feedback takes the form of (3.2), the agent is allowed to collect only *one* measurement at a certain point at its disposal at every sampling instance τ_k. Then (2.1) may include the one-measurement SPSA under further assumptions. Specifically, $y_k(\cdot)$ will be evaluated at the design point $\hat{\theta}_k + c_k\boldsymbol{\Delta}_k$, where $\boldsymbol{\Delta}_k$ is a p-dimensional random vector with zero-mean satisfying conditions listed in [106, Sect. 7.4], and c_k is a small positive number strictly bounded away from zero. For one-measurement SPSA, $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ in (2.1) will be substituted by $\hat{\boldsymbol{g}}_k^{\mathrm{SP1}}(\hat{\theta}_k)$ computed as in [103]:

$$\hat{\boldsymbol{g}}_k^{\mathrm{SP1}}(\hat{\theta}_k) \equiv \frac{y_k(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k)}{c_k}\boldsymbol{\Delta}_k^{-1}, \tag{3.4}$$

  where $\mathrm{SP1}$ in the superscript is short for "simultaneous perturbation with one-measurement" [106, Sect. 7.3].

  If the agent is allowed to collect only *two* measurements, then (2.1) may include the two-measurement SPSA under further assumptions. Specifically, the agent can evaluate $y_k(\cdot)$ at two design points $\hat{\theta}_k + c_k\boldsymbol{\Delta}_k$ and $\hat{\theta}_k - c_k\boldsymbol{\Delta}_k$, where $\boldsymbol{\Delta}_k$ satisfies

the same condition mentioned above. For two-measurement SPSA, $\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ in (2.1) will be replaced by $\hat{\boldsymbol{g}}_k^{\mathrm{SP2}}(\hat{\boldsymbol{\theta}}_k)$ discussed in Subsection 2.1.2, except that $y(\cdot)$ in (2.4) is substituted by $y_k(\cdot)$ that depends on k as (3.2).

- When the feedback takes the form of (3.3), the agent is allowed to collect only *one* measurement of $\boldsymbol{Y}_k(\cdot)$ evaluated at the decision point $\hat{\boldsymbol{\theta}}_k$. In this case, (2.1) may include the well-known stochastic gradient algorithm proposed in [91]. For SGD, $\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ is a direct noisy gradient measurement as in (2.5), except that the $\boldsymbol{g}(\cdot)$ on the r.h.s. of (2.5) now has a k-dependence.

With a slight abuse of notation $\boldsymbol{\beta}_k(\hat{\boldsymbol{\theta}}_k)$ and $\boldsymbol{\xi}_k(\hat{\boldsymbol{\theta}}_k)$ appearing in (2.2), we express $\hat{\boldsymbol{g}}_k(\cdot)$ generically as below to facilitate later discussion:

$$\begin{aligned}\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) &= \left.\frac{\partial f_k(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_k} + \boldsymbol{\beta}_k(\hat{\boldsymbol{\theta}}_k) + \boldsymbol{\xi}_k(\hat{\boldsymbol{\theta}}_k) \\ &\equiv \boldsymbol{g}_k(\hat{\boldsymbol{\theta}}_k) + \boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_k) \end{aligned} \tag{3.5}$$

where the gradient function $\boldsymbol{g}_k(\boldsymbol{\theta}) \equiv \partial f_k(\boldsymbol{\theta})/\partial \boldsymbol{\theta}$, the error term $\boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_k)$ subsumes both the bias term $\boldsymbol{\beta}_k(\hat{\boldsymbol{\theta}}_k)$ and the noise term $\boldsymbol{\xi}_k(\hat{\boldsymbol{\theta}}_k)$. Note that $\boldsymbol{g}_k(\cdot)$ on the r.h.s. of (3.5) has k-dependence, whereas $\boldsymbol{g}(\cdot)$ on the r.h.s. of (2.2) does not. Moreover, the function $\boldsymbol{g}(\cdot)$ for root-finding purposes in Chapter 2 may or may not be a gradient of an underlying loss function.

3.1.3 Distinction Relative to Other Finite-Sample Analysis

Among the finite-sample performance analysis, [117] is derived under a similar setup and used a comparable metric. We point out several differences between [117] and our work.

- Ref. [117] assumes that multiple, e.g., N_k, sequential measurements of (3.3) can be gathered at each sampling time instant τ_k. Specifically, N_k grows inversely proportional to the desired accuracy, which may be expensive as mentioned towards the end of [117, Sect. 2.1]. By "sequential" we mean that the N_k observations at time τ_k have to be carried out sequentially. That is, the $(i+1)$th observation depends on the i-th observation for $1 \leq i < N_k$. Such a setting may be valid if the underlying time-varying system is changing very slowly or if the experimenter has a nearly unlimited amount of computation power and does not get penalized for frequent observations.

  In contrast, we do not allow the sequential observations at each sampling time τ_k and discourages excessive observations at each iteration. We consider few parallel measurements at each τ_k," e.g., $N_k = 1$ or 2, in order to readily adapt to changing conditions. By "parallel" we mean that the evaluations at the two design points at time τ_k can be collected simultaneously—one does not depend on the computation of another one.

- Ref. [117] assumes that the immediate feedback is in the form of (3.3) only, whereas our work allows the feedback to take the form of either (3.2) or (3.3). Moreover, [117] assumes that the noisy gradient measurement is an unbiased estimator of the true gradient; i.e., the $\boldsymbol{e}_k$ in (3.5) under their setting is mean-zero, whereas we allow $\boldsymbol{e}_k$ in (3.5) to have a nonzero mean in general.

- In both tracking criteria proposed in [117], the randomness in θ_k^* is not considered, and hence the randomness in f_k is not allowed. On the contrary, our tracking performance result in Section 3.3 allows for some randomnesses in θ_k^*.

- Ref. [117] implicitly assumes that the selected gain will enable the estimate to keep track of the moving target, and does not unveil their details in the gain selection. In contrast, we provide some practical guidance in gain selection.

3.2 Model Assumptions

We now state the assumptions required for later derivations. Throughout our discussion, the norm imposed on a vector is the Euclidean norm, and the norm imposed on a matrix is the matrix spectral norm, which is the matrix norm compatible with the Euclidean vector norm. The following assumptions are in parallel with the statistical set of conditions for the strong convergence in [15, Sect. 2] and [106, Sect. 4.3], except for the non-decaying gain adapted for the extra restrictions on the drift and the nonstationarity explained in the next section.

*Assumption A.*1 (Error Term Has Bounded Second Moment). There exists a finite number $\mathscr{M}_k \equiv \sqrt{\sup_{\boldsymbol{\theta}\in\mathbb{R}^p} \mathbb{E}[\|\boldsymbol{e}_k(\boldsymbol{\theta})\|^2]}$ for each $k \in \mathbb{N}$.

*Assumption A.*2 (Strong Convexity). The instantaneous loss function $f_k(\cdot) \in C^1(\mathbb{R}^p \mapsto \mathbb{R})$ and strongly convex for all $k \in \mathbb{N}$. Moreover, $\mathscr{C}_k$ is the largest positive number such that $(\boldsymbol{\theta}-\boldsymbol{\zeta})^T(\boldsymbol{g}_k(\boldsymbol{\theta})-\boldsymbol{g}_k(\boldsymbol{\zeta})) \geq \mathscr{C}_k\|\boldsymbol{\theta}-\boldsymbol{\zeta}\|^2$ holds for all $\boldsymbol{\theta}, \boldsymbol{\zeta} \in \mathbb{R}^p$, where $\boldsymbol{g}_k(\boldsymbol{\theta}) = \partial f_k(\boldsymbol{\theta})/\partial\boldsymbol{\theta}$.

*Assumption A.*3 (Smoothness). For each $k \in \mathbb{N}$, $\mathscr{L}_k$ is the smallest positive number such that $\boldsymbol{g}_k(\cdot) \in C^0(\mathbb{R}^p \mapsto \mathbb{R}^p)$ is $\mathscr{L}_k$-Lipschitz.

Assumption A. 4 (Bounded Variation). There exists a finite number $\mathscr{B}_k \equiv \sqrt{\mathbb{E}(\|\boldsymbol{\theta}^*_{k+1} - \boldsymbol{\theta}^*_k\|^2)}$ for each $k \in \mathbb{N}$. It reduces to $\mathscr{B}_k = \sqrt{(\|\boldsymbol{\theta}^*_{k+1} - \boldsymbol{\theta}^*_k\|^2)}$ if the sequence $\{\boldsymbol{\theta}^*_k\}$ is deterministic.

Remark 4. In addition to Remark 3, we reiterate that the randomness in $\boldsymbol{\theta}^*_k$ is *not* taken into consideration while formulating $f_k(\cdot)$, but $\boldsymbol{\theta}^*_k$ itself is allowed to vary stochastically.

To ease the upcoming discussion, denote the ratio $\mathscr{R}_k \equiv \mathscr{L}_k/\mathscr{C}_k$, where $\mathscr{L}_k$ is defined in A.3 and $\mathscr{C}_k$ is defined in A.2. The following subsections provide additional explanations on the validity of the aforementioned assumptions.

3.2.1 Estimation of Parameters in Assumptions Will Not be Considered

Note that this chapter aims to show the tracking capability of SA algorithms (2.1) with non-decaying gains applied to the time-varying problem setup (3.1). Furthermore, we are interested in the scenario where only a few (one or two) noisy observations pertaining to $f_k(\cdot)$ are revealed only at time instance k, and the actual time elapsed between two

consecutive sampling instances $(\tau_{k+1} - \tau_k)$ is bounded from below. Under such a setting, at every sampling instance τ_k, the agent obtains a limited amount of corrupted information regarding $f_k(\cdot)$. Resultingly, we do not expect that there exists an efficient strategy to estimate $\mathscr{M}_k$, $\mathscr{C}_k$, $\mathscr{L}_k$, and $\mathscr{B}_k$ in an online fashion. Nonetheless, the assumed availability of these parameters does not nullify the deliverables of this chapter in demonstrating the tracking capability of SA algorithms.

It was pointed out in Subsection 3.1.3 that although [117] handles the estimation in part, [117] is based upon a different setup. Namely, they assume that as many *sequential* estimates as needed can be gathered at each sampling instance τ_k, whereas our setup requires one single observation or two *parallel* ones. Furthermore, the estimation of the lower-bound of $\mathscr{C}_k$ based on (3.11) and the estimation of the upper-bound of $\mathscr{L}_k$ based on (3.12) could be largely non-informative regarding the actual value of $\mathscr{C}_k$ and $\mathscr{L}_k$.

3.2.2 Relation With Online Learning Literature

Connection. The sequential SO set up in Section 3.1 can be interpreted in the prototypical decision-making framework. The agents are viewed as learners and targets as adversaries. At each sampling instance τ_k, the online learner selects an action $\hat{\theta}_k$ that belongs to some convex compact action set $\boldsymbol{\Theta} \subsetneq \mathbb{R}^p$ and incurs a cost $f_k(\hat{\theta}_k)$, where $f_k(\cdot) : \mathbb{R}^p \mapsto \mathbb{R}$ is an unknown convex cost function selected by the adversary. In response to the agent's action, the adversary also reveals inexact feedback to the learner.

Distinctions. Different from the constraint that the variable θ belongs to a compact domain $\boldsymbol{\Theta}$, we consider the situation where the objective function is strongly convex and $\theta \in \mathbb{R}^p$. As opposed to the result that allows the loss $f_k(\cdot)$ to be adversarial w.r.t. the selected action $\hat{\theta}_k$, we consider the case where $f_k(\cdot)$ is deterministic or maybe random, but has to be autonomous. By "autonomous" we mean that the values of the estimates $\hat{\theta}_k$ do not affect the underlying evolution of θ_k^*. Contrary to the strong requirement that "all the loss functions $f_k(\cdot)$ have uniformly bounded gradients," we consider a weaker assumption as in A.3. Different from the goal of bounding the worst-case performance of the best estimators only through the regret formulation [47], we are interested in the tracking accuracy, i.e., controlling the error $\|\hat{\theta}_k - \theta_k^*\|$ at each time τ_k. Moreover, the regret $\text{Reg}_K \equiv \sum_{k=1}^K [f_k(\hat{\theta}_k) - f_k(\theta_k^*)]$ is minimized, where K is the horizon over which we implement the recursive scheme (2.1), under the condition that there exists a bound on the total variations of the gradients over the horizon K. Admittedly, if A.2 is satisfied, then the bound on $\sum_{k=2}^K \sup_{\theta\in\boldsymbol{\Theta}} \|\boldsymbol{g}_k(\theta) - \boldsymbol{g}_{k-1}(\theta)\|^2$ implies the bound on $\sum_{k=2}^K \|\theta_k^* - \theta_{k-1}^*\|^2$. The converse is not true. In contrast, we only impose A.4 and seek to maintain a certain tracking accuracy at each time instant. Note that we do not get into online prediction where the regret along the path is minimized. Instead, we are only interested in minimizing the most current estimation error. Another reason is that the error bound developed for many algorithms therein requires knowing the functional variation $\sum_{k=2}^K \sup_{\theta\in\boldsymbol{\Theta}} |f_k(\theta) - f_{k-1}(\theta)|$ in advance, which is typically unavailable.

3.2.3 Error Form Allowing Many SA Algorithms

Let us emphasize that A.1 allows for $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ to be a biased estimate for $\boldsymbol{g}_k(\hat{\theta}_k)$; i.e., the error term $\boldsymbol{e}_k$ in (3.5) can have a nonzero mean. Furthermore, A.1 enables recursion (2.1) to subsume a broad class of SA algorithms, including the three important cases mentioned in Section 3.1.2.

For one-measurement and two-measurement SPSA, A.1 is readily satisfied when the following holds: (1) $\boldsymbol{\Delta}_k$ is generated by Monte Carlo under the conditions of independence, symmetry, and finite inverse moments [101] and (2) there exists[2] c_{lower} such that $0 < c_{\text{lower}} < c_k$ for all k.

For R-M setting, A.1 is immediately met because zero-mean and bounded-variance $\boldsymbol{e}_k(\cdot)$ is a special case of A.1 as per [106, A.3 and A.4 on p. 106].

3.2.4 Global and Local Convexity Parameter

A direct consequence of A.2 is the existence and uniqueness of the optimizer θ_k^*. Moreover, $\boldsymbol{g}_k(\theta_k^*) = \mathbf{0}$ becomes a necessary and sufficient condition in determining θ_k^*, and it will be used in proving the upcoming Lemma 3.3.1. Admittedly, there is a class of nonconvex problems in which A.2 fails to hold. Nonetheless, A.2 is still valid in many fundamental problems such as regularized regression and many others in [10]. An incomplete list is given below.

[2]In the time-varying scenario, both a_k and c_k have to be strictly positive for the recursive SA algorithm to be able to track the moving target. In FDSA or SPSA, the gain sequence controlling the perturbation magnitude is set to be strictly bounded away from zero for stability, despite that in theory a decaying gain can wash out the bias of the gradient approximation as an estimator of the true gradient.

- Suppose that the loss function $f(\cdot)$ is in the empirical risk function (ERF) form. For instance, given data pairs $(\boldsymbol{x}_i, z_i)$ where the covariate $\boldsymbol{x}_i$ will be mapped by a function $\boldsymbol{\Phi}(\cdot)$ to the feature space $\mathbb{R}^p$, then the loss function $f(\cdot)$ can be formed as $f(\theta) = n^{-1}\sum_{i=1}^{n}\ell\left(z_i, \theta^T\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right)\right)$, where $\ell(\cdot,\cdot)$ denotes either the squared-loss or zero-one loss, and the input has a nonsingular sample covariance matrix $n^{-1}\sum_{i=1}^{n}\{[\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right)]^T\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right)\}$. Such a loss function $f(\cdot)$ satisfies A.2.

- Suppose that loss function $f(\cdot)$ is the sum of $n^{-1}\sum_{i=1}^{n}\ell\left(z_i, \theta^T\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right)\right)$ and a regularization term $\mathscr{C}\|\theta\|^2/2$, where $\boldsymbol{\Phi}(\cdot)$ maps the input $\boldsymbol{x}_i$ to the intended feature space. Then A.2 is satisfied.

- Suppose that the loss function $f(\cdot)$ is the expected least-squares written as $f\left(\theta\right) = \mathbb{E}\left[z - \theta_{\text{true}}^T\boldsymbol{\Phi}(x)\right]^2/2$, where the expectation is taken over the joint-distribution of $(\boldsymbol{x}, z)$. When $\mathbb{E}\{[\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right)]^T\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right)\} \succ \mathscr{C}\boldsymbol{I}_p$, the loss function satisfies A.2.

Note that A.2 can be relaxed to local strong convexity, as the proofs in the upcoming section require local convexity only. Namely, the $\mathscr{C}_k$ in A.2 can be the largest positive number such that $(\theta - \zeta)^T(\boldsymbol{g}_k(\theta) - \boldsymbol{g}_k(\zeta)) \geq \mathscr{C}_k\|\theta - \zeta\|^2$ for θ, ζ in a small neighborhood around $\hat{\theta}_k$. However, we do not intend to dwell on the "multiple-minimizers" setting. The rationale and the tracking capability of non-decaying gain SA under such a scenario require separate consideration.

3.2.5 Global- and Local-Lipschitz Continuity

Note that A.3 is more lenient than the uniform boundedness of $\boldsymbol{g}_k(\cdot)$ for all θ uniformly across k appearing in [85, Sect. 6.3], [9], and many others. In fact, A.3 can be met in the sense that it is implied by other smoothness conditions that are used in local convergence theorems and are often satisfied in practice [80, p. 39 and Chaps 6–7].

Let us provide an example in machine learning applications where A.3 is satisfied. Suppose that the loss function $f(\cdot)$ is in the ERF form, i.e., $f(\theta) = n^{-1}\sum_{i=1}^{n}\ell\left(z_i, \theta^T\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right)\right)$ for some squared- or 0-1 loss function $\ell(\cdot,\cdot)$, where the data pairs $(\boldsymbol{\Phi}(\boldsymbol{x}_i), z_i)$ are all bounded. The Hessian of $f(\cdot)$ is approximately the sample covariance matrix computed as $n^{-1}\sum_{i=1}^{n}[\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right)\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right)]$ for large n, hence A.3 is satisfied.

Arguably, A.3 does not hold even for a scalar-valued univariate function $g_k(\theta)$ with $g_k(\cdot)$ being a second- or higher-order polynomial function or the multiplicative-inverse function defined over $\theta \in \mathbb{R}$. The following two observations help alleviate the concern regarding its appropriateness:

- In real-world applications, the parameter θ is typically subject to physical restrictions or other technical constraints. For instance, if θ is confined within a closed and bounded region $\boldsymbol{\Theta} \subsetneq \mathbb{R}^p$, then a finite $\mathscr{L}_k$ within $\boldsymbol{\Theta}$ is attainable.

- In the upcoming proof, we can effectively replace the global smoothness by local smoothness $\mathscr{L}_k^{(\hat{\theta}_k, \theta_k^*)}$, the smallest number such that $\|\boldsymbol{g}_k(\theta_1) - \boldsymbol{g}_k(\theta_2)\| \leq$

$\mathscr{L}_k^{(\hat{\theta}_k, \theta_k^*)} \|\theta_1 - \theta_2\|$ holds for any θ_1, θ_2 in a ball centered at $\hat{\theta}_k$ with radius of $\|\hat{\theta}_k - \theta_k^*\|$.

In summary, both A.2 and A.3 can be weakened if a priori knowledge of the domain of the optimizers, denoted by $\boldsymbol{\Theta}$, is known. If so, we can concentrate on functions that meet A.2 and A.3 for $\theta \in \boldsymbol{\Theta} \subsetneq \mathbb{R}^p$ and adapt the following proof for algorithm (2.6) readily. Nonetheless, the analysis in Section 3.3 reveals that the estimate $\hat{\theta}_k$ will stay close to θ_k^* with appropriate initialization and gain selection, and therefore we only require A.2 and A.3 to be valid locally.

3.2.6 Interpreting Ratio of $\mathscr{L}_k$ and $\mathscr{C}_k$

Note that for $\mathscr{C}_k$ in A.2 and $\mathscr{L}_k$ in A.3 to be well-defined, we only need $f_k(\cdot)$ be in $C^1(\mathbb{R}^p \mapsto \mathbb{R})$, i.e., continuously differentiable, as stated in A.2.

To provide a better intuition behind $\mathscr{R}_k$, let us further assume [106, Assumption B.5" on p. 183], i.e., the loss function $f_k(\cdot)$ is in $C^2(\mathbb{R}^p \mapsto \mathbb{R})$ and is bounded on $\mathbb{R}^p$. Let us denote $\boldsymbol{H}_k(\cdot)$ as the Hessian of $f_k(\cdot)$, which is guaranteed to be square and positive-definite by A.2. By Taylor's Theorem for multivariate vector-valued function, we know $\mathscr{C}_k$ in A.2 becomes $\inf_{\theta \in \mathbb{R}^p} \lambda_{\min}(\boldsymbol{H}_k(\theta))$, and $\mathscr{L}_k$ in A.3 equals $\sup_{\theta \in \mathbb{R}^p} \lambda_{\max}(\boldsymbol{H}_k(\theta))$. Note that both the inf and the sup are attainable under [106, Assumption B.5" on p. 183] as both $\lambda_{\min}(\boldsymbol{H}_k(\theta))$ and $\lambda_{\max}(\boldsymbol{H}_k(\theta))$ are continuous functions of θ. Therefore, when $\boldsymbol{H}_k(\cdot)$ exists and satisfies certain smoothness conditions, $\mathscr{R}_k$ can be interpreted to be an upper

bound of the condition number of the Hessian because:

$$\mathscr{R}_k = \frac{\mathscr{L}_k}{\mathscr{C}_k} = \frac{\sup_{\theta\in\mathbb{R}^p} \lambda_{\max}(\boldsymbol{H}_k(\theta))}{\inf_{\theta\in\mathbb{R}^p} \lambda_{\min}(\boldsymbol{H}_k(\theta))} \geq \sup_{\theta} \frac{\lambda_{\max}(\boldsymbol{H}_k(\theta))}{\lambda_{\min}(\boldsymbol{H}_k(\theta))} = \sup_{\theta} \left(\mathrm{cond}(\boldsymbol{H}_k(\theta))\right). \quad (3.6)$$

3.2.7 Parameter Variations and Error Bounds

Note that the model for $\{\theta_k^*\}$ is autonomous because updating $\hat{\theta}_k$ by (2.1) has no effect on the true parameter θ_k^*. Intuitively, A.4 is imposed to capture the fact that the sequence (3.1) is changing "slowly," yet it does not exclude abrupt changes as long as the corresponding probability is small. Overall, the expected change of the optimal parameter between every two consecutive time instants is modest.

Within the classical literature on linear models, [34, 37] obtained upper bounds of the limiting-time mean-square error (the deviation of the estimated value of the parameter from the actual value) by assuming only that the speed of variation of the true system is bounded by some deterministic constant. We port the idea over to the general nonlinear models.

In physical application to moving objects, A.4 effectively sets a maximum speed of the target, which is generally reasonable, given the physical constraints of motion. If the target's position is denoted by θ_k^* and it is moving at a constant speed, then no randomness arises in the sequence in $\{\theta_k^*\}$ and $\sum_{k=1}^{K} \|\theta_k^* - \theta_{k-1}^*\|$ is $O(\tau_K - \tau_0)$. As a practical example, consider a target that continues to move at a constant speed in an adversarial manner. Likewise, we may consider the scenario when the errors are unpredictably random with the bounded second moment. We point out that [117,

Sect. 2.2] provides other justifications for A.4. Therefore, for the target tracking setting, it makes sense to characterize the parameter variations using the path length $\sum_{k=1}^{K} \|\theta_k^* - \theta_{k-1}^*\|$ for some sequence of parameter values $\{\theta_k^*\}$. Alternative measures might include functional variation $\sum_{k=1}^{K} \sup_{\theta\in\Theta} |f_k(\theta) - f_{k-1}(\theta)|$ and the gradient variation $\sum_{k=1}^{K} \sup_{\theta\in\Theta} \|\boldsymbol{g}_k(\theta) - \boldsymbol{g}_{k-1}(\theta)\|^2$, which are usually unavailable in advance.

3.3 Tracking Performance Guarantee

This section characterizes the tracking performance $\|\hat{\theta}_k - \theta_k^*\|$ of recursion (2.1) with non-diminishing gain, where $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ can take either of the representations (3.4) and (2.5). When the target is perpetually varying, it is impossible for the agent to further reduce its distances from the target beyond a certain value. Hence, we assume the necessary assumptions in Section 4.1.3 to facilitate our error bound analysis. Note that the values of $\mathscr{M}_k$, $\mathscr{L}_k$, and $\mathscr{B}_k$ are the smallest possible positive reals such that the assumptions A.1, A.3, and A.4 are valid, and the value of $\mathscr{C}_k$ is the largest possible real such that the assumption A.2 is legitimate. The analysis here is built upon the basis that $\mathscr{M}_k$ in A.1, $\mathscr{C}_k$ in A.2, $\mathscr{L}_k$ in A.3, and $\mathscr{B}_k$ in A.4 are available to the agent.

3.3.1 Supporting Lemmas

Lemma 3.3.1 provides some inequalities that immediately follow from the assumptions stated in Section 3.2. They will be used in the upcoming subsection.

Lemma 3.3.1. *For a loss function f_k satisfying A.2 and A.3, the following inequalities hold for all $\theta \in \mathbb{R}^p$:*

$$\mathscr{C}_k\|\theta - \theta_k^*\|^2/2 \leq f_k(\theta) - f_k(\theta_k^*) \leq \mathscr{L}_k\|\theta - \theta_k^*\|^2/2, \tag{3.7}$$

$$2\mathscr{C}_k\left[f_k(\theta) - f_k(\theta_k^*)\right] \leq \|\boldsymbol{g}_k(\theta)\|^2 \leq 2\mathscr{L}_k\left[f_k(\theta) - f_k(\theta_k^*)\right], \tag{3.8}$$

$$f_k(\theta) - f_k(\theta_k^*) \leq \boldsymbol{g}_k(\theta)^T(\theta - \theta_k^*), \tag{3.9}$$

$$\mathscr{L}_k \geq \mathscr{C}_k > 0. \tag{3.10}$$

Proof of Lemma 3.3.1. Given A.2, we know that for any $\theta, \zeta \in \mathbb{R}^p$:

$$f_k(\theta) \geq f_k(\zeta) + \left[\boldsymbol{g}_k(\zeta)\right]^T(\theta - \zeta) + \frac{\mathscr{C}_k}{2}\|\theta - \zeta\|^2. \tag{3.11}$$

Let $\zeta = \theta_k^*$ in (3.11) and invoke A.2. We then have $f_k(\theta) \geq f_k(\theta_k^*) + \mathscr{C}_k\|\theta - \theta_k^*\|^2/2$ and, therefore, the first inequality of (3.7) holds.

By A.3 and the mean-value theorem [93], we know that for any $\theta, \zeta \in \mathbb{R}^p$:

$$\begin{aligned}
f_k(\theta) &= f_k(\zeta) + \int_0^1 \left[\boldsymbol{g}_k(\zeta + t(\theta - \zeta))\right]^T(\theta - \zeta)\, dt \\
&= f_k(\zeta) + \left[\boldsymbol{g}_k(\zeta)\right]^T(\theta - \zeta) + \int_0^1 \left[\boldsymbol{g}_k(\zeta + t(\theta - \zeta)) - \boldsymbol{g}_k(\zeta)\right]^T(\theta - \zeta)dt \\
&\leq f_k(\zeta) + \left[\boldsymbol{g}_k(\zeta)\right]^T(\theta - \zeta) + \int_0^1 \mathscr{L}_k\|t(\theta - \zeta)\|\|\theta - \zeta\|dt \\
&= f_k(\zeta) + \left[\boldsymbol{g}_k(\zeta)\right]^T(\theta - \zeta) + \frac{\mathscr{L}_k}{2}\|\theta - \zeta\|^2.
\end{aligned} \tag{3.12}$$

Let $\boldsymbol{\zeta} = \boldsymbol{\theta}_k^*$ in (3.12) and invoke A.2, we then have $f_k(\boldsymbol{\theta}) \leq f_k(\boldsymbol{\theta}_k^*) + \mathscr{L}_k\|\boldsymbol{\theta} - \boldsymbol{\theta}_k^*\|^2/2$. Hence, the second inequality of (3.7) holds.

Note that for every $\boldsymbol{\theta} \in \mathbb{R}^p$, (3.11) holds. Let us deem both sides of (3.11) as two functions of $\boldsymbol{\theta}$. By definition, the minimum of the l.h.s. is achieved by $\boldsymbol{\theta} = \boldsymbol{\theta}_k^*$. The minimizer of the r.h.s., which is a quadratic function of $\boldsymbol{\theta}$, is given by $\boldsymbol{\theta} = \boldsymbol{\zeta} - \mathscr{C}_k\boldsymbol{g}_k(\boldsymbol{\zeta})$. Therefore,

$$\begin{aligned} f_k(\boldsymbol{\theta}_k^*) &\geq \left\{ f_k(\boldsymbol{\zeta}) + [\boldsymbol{g}_k(\boldsymbol{\zeta})]^T(\boldsymbol{\theta} - \boldsymbol{\zeta}) + \frac{\mathscr{C}_k}{2}\|\boldsymbol{\theta} - \boldsymbol{\zeta}\|^2 \right\}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_k^*} \\ &\geq \left\{ f_k(\boldsymbol{\zeta}) + [\boldsymbol{g}_k(\boldsymbol{\zeta})]^T(\boldsymbol{\theta} - \boldsymbol{\zeta}) + \frac{\mathscr{C}_k}{2}\|\boldsymbol{\theta} - \boldsymbol{\zeta}\|^2 \right\}\Big|_{\boldsymbol{\theta}=\boldsymbol{\zeta}-\mathscr{C}_k^{-1}\boldsymbol{g}_k(\boldsymbol{\zeta})} \\ &= f_k(\boldsymbol{\zeta}) - \frac{1}{2\mathscr{C}_k}\|\boldsymbol{g}_k(\boldsymbol{\zeta})\|^2. \end{aligned}$$

Hence, the first inequality in (3.8) holds.

Note that (3.12) holds for any $\boldsymbol{\theta}, \boldsymbol{\zeta} \in \mathbb{R}^p$ so we have:

$$\begin{aligned} f_k(\boldsymbol{\theta}_k^*) &\leq \min_{\boldsymbol{\theta}} \left\{ f_k(\boldsymbol{\zeta}) + [\boldsymbol{g}_k(\boldsymbol{\zeta})]^T(\boldsymbol{\theta} - \boldsymbol{\zeta}) + \frac{\mathscr{L}_k}{2}\|\boldsymbol{\theta} - \boldsymbol{\zeta}\|^2 \right\} \\ &= \left\{ f_k(\boldsymbol{\zeta}) + [\boldsymbol{g}_k(\boldsymbol{\zeta})]^T(\boldsymbol{\theta} - \boldsymbol{\zeta}) + \frac{\mathscr{L}_k}{2}\|\boldsymbol{\theta} - \boldsymbol{\zeta}\|^2 \right\}\Big|_{\boldsymbol{\theta}=\boldsymbol{\zeta}-\mathscr{L}_k^{-1}\boldsymbol{g}_k(\boldsymbol{\zeta})} \\ &= f_k(\boldsymbol{\zeta}) - \frac{1}{2\mathscr{L}_k}\|\boldsymbol{g}_k(\boldsymbol{\zeta})\|^2. \end{aligned}$$

Hence, the second inequality in (3.8) holds.

Eq. (3.9) follows from A.2; specifically, $f_k(\boldsymbol{\theta}_k^*) \geq f_k(\boldsymbol{\theta}) + \boldsymbol{g}_k(\boldsymbol{\theta})^T(\boldsymbol{\theta}_k^* - \boldsymbol{\theta})$.

Eq. (3.10) can be readily obtained by comparing (3.11) and (3.12). □

We also present Lemma 3.3.2 and Lemma 3.3.3 here in anticipation of handling the upcoming recursive inequality.

Lemma 3.3.2. *Let $\{x_k\}$ be a sequence of scalars such that $|x_k| \leq 1$ for all k. Then for $1 \leq j \leq k$ and $k \geq 1$, we have*

$$\sum_{i=j}^{k} \prod_{l=i+1}^{k} (1 - x_l)x_i = 1 - \prod_{i=j}^{k}(1 - x_i). \tag{3.13}$$

We take the tradition that the cumulative product equals one if the starting index is no smaller than the ending index.

Proof of Lemma 3.3.2. Note that

$$\prod_{l=i+1}^{k} (1 - x_l) - \prod_{l=i}^{k}(1 - x_l) = \prod_{l=i+1}^{k} \{(1 - x_l)\,[1 - (1 - x_i)]\} = \prod_{l=i+1}^{k} [(1 - x_l)x_i]\,.$$

Thus,

$$\prod_{l=i+1}^{k} [(1 - x_l)x_i] = \left[\prod_{l=i+1}^{k} (1 - x_l) - \prod_{l=j}^{k}(1 - x_l)\right] - \left[\prod_{l=i}^{k}(1 - x_l) - \prod_{l=j}^{k}(1 - x_l)\right].$$

Summing the above equation over i from j to k on the r.h.s. collapses to yield equation (3.13). □

Lemma 3.3.3. *Let $\{x_k\}$ be a scalar sequence such that $0 < \inf_k x_k \leq \sup_k x_k < X < \infty$. Let $\{z_k\}$ be a sequence such that $0 \leq z_k < 1$. Define $\nu_{j,k} \equiv (1 - z_j) \prod_{i=j+1}^{k} z_j$ for $j < k$.*

Then

$$\limsup_k \sum_{j=1}^{k} \nu_{j,k} x_j \leq \limsup_k x_k. \tag{3.14}$$

Proof of Lemma 3.3.3. Denote $\tilde{X} = \limsup_k x_k$. Then for any $\varepsilon > 0$, there exists a finite k_0 such that $x_k < \tilde{X} + \varepsilon$ for all $k > k_0$. For such indices, we have

$$\sum_{j=1}^{k} \nu_{j,k} x_k < X \sum_{j=1}^{k_0} \nu_{j,k} + \tilde{X} + \varepsilon.$$

where the inequality follows from the result in Lemma 3.3.2. Furthermore, the term $\sum_{j=1}^{k_0} \nu_{j,k}$ goes to zero as $k \to \infty$. That is, there exists a finite $k_1 > k_0$ such that the term $\sum_{j=1}^{k_0} \nu_{j,k}$ remains smaller than ε/X for all $k > k_1$.

Therefore, for sufficiently large k, we have

$$\sum_{j=1}^{k} \nu_{j,k} x_k < \tilde{X} + 2\varepsilon.$$

Given that $\varepsilon > 0$ is arbitrary, our desired result (3.14) holds. □

The main theorems in this section pertain to a positive slack variable q_k whose allowable domain depends on $\mathscr{R}_k$. For brevity, the dependence of q_k's domain on $\mathscr{R}_k$ will be suppressed wherever no confusion is introduced. The slack variable q_k can be effectively viewed as a hyper-parameter, which shall be picked based on the smoothness parameter $\mathscr{L}_k$ and the strong convexity parameter $\mathscr{C}_k$, before selecting the gain sequence a_k. This is natural as $\mathscr{R}_k$ pertains to the curvature information of the loss function $f_k(\cdot)$ presented

in (3.6) when additional smoothness condition [107, Assumption B.5" on p. 183] is met. Let us present several lemmas to control the slack variable q_k to better serve the upcoming proofs on the tracking performance.

Lemma 3.3.4. *If $\mathscr{R}_k > 1$ (i.e., $\mathscr{L}_k > \mathscr{C}_k > 0$), we have $\mathscr{C}_k^4 (q+2)^2 - 4\mathscr{L}_k^2\mathscr{C}_k^2 (q+1) \leq 0$ for any:*

$$0 < q \leq q_{k,1}(\mathscr{R}_k) \equiv 2(\mathscr{R}_k^2 - 1) + 2\mathscr{R}_k\sqrt{\mathscr{R}_k^2 - 1}. \tag{3.15}$$

We will rewrite $q_{k,1}(\mathscr{R}_k)$ as $q_{k,1}$ wherever convenient.

Proof of Lemma 3.3.4. Define $h_{k,1}(q) \equiv (q+2)^2 - 4\mathscr{R}_k^2 (q+1) = q^2 + 4(1-\mathscr{R}_k^2) q + 4(1-\mathscr{R}_k^2)$, and it is a quadratic function of q. The determinant[3] of the quadratic function $h_{k,1}(\cdot)$ is $\Delta_{k,1} \equiv 16(1-\mathscr{R}_k^2)^2 - 16(1-\mathscr{R}_k^2) = 16\mathscr{R}_k^2(\mathscr{R}_k^2 - 1)$. If $\mathscr{R}_k > 1$, we have $\Delta_{k,1} > 0$. The two real roots of $h_{k,1}(\cdot)$ are $2(\mathscr{R}_k^2 - 1) - 2\mathscr{R}_k\sqrt{\mathscr{R}_k^2 - 1}$ and $2(\mathscr{R}_k^2 - 1) + 2\mathscr{R}_k\sqrt{\mathscr{R}_k^2 - 1} (\equiv q_{k,1})$ respectively. Given that the sum of the two real roots is $4(\mathscr{R}_k^2 - 1) > 0$, and the product of the two real roots is $4(1-\mathscr{R}_k^2) < 0$, we know that the smaller root is negative and the larger root $q_{k,1}$ is positive. Therefore, when $\mathscr{R}_k > 1$, $\mathscr{C}_k^4 (q+2)^2 - 4\mathscr{L}_k^2\mathscr{C}_k^2 (q+1) \leq 0$ is nonpositive for any q satisfying (3.15). □

Lemma 3.3.5. *If $1 \leq \mathscr{R}_k \leq (1+\sqrt{5})/2$, we have $\mathscr{C}_k^4 (q+2)^2 - 4\mathscr{C}_k^2\mathscr{L}_k(\mathscr{L}_k - \mathscr{C}_k)(q+1) > 0$ for any $q > 0$. If $\mathscr{R}_k > (1+\sqrt{5})/2$, we have*

[3]For a general quadratic function $ax^2 + bx + c$ of x, its determinant is defined to be $\Delta \equiv b^2 - 4ac$.

$\mathscr{C}_k^4 (q+2)^2 - 4\mathscr{C}_k^2 \mathscr{L}_k (\mathscr{L}_k - \mathscr{C}_k)(q+1) > 0$ *for any:*

$$q > q_{k,2}(\mathscr{R}_k) \equiv 2(\mathscr{R}_k^2 - \mathscr{R}_k - 1) + 2\sqrt{\mathscr{R}_k(\mathscr{R}_k - 1)(\mathscr{R}_k^2 - \mathscr{R}_k - 1)}. \tag{3.16}$$

We will rewrite $q_{k,2}(\mathscr{R}_k)$ as $q_{k,2}$ wherever convenient.

Proof of Lemma 3.3.5. Define $h_{k,2}(q) \equiv (q+2)^2 - 4\mathscr{R}_k(\mathscr{R}_k - 1)(q+1) = q^2 + 4(1 + \mathscr{R}_k - \mathscr{R}_k^2)q + 4(1 + \mathscr{R}_k - \mathscr{R}_k^2)$. The determinant of the quadratic function $h_{k,2}(\cdot)$ is $\Delta_{k,2} = 16(1 + \mathscr{R}_k - \mathscr{R}_k^2)^2 - 16(1 + \mathscr{R}_k - \mathscr{R}_k^2) = 16\mathscr{R}_k(\mathscr{R}_k - 1)(\mathscr{R}_k^2 - \mathscr{R}_k - 1)$.

If $\mathscr{R}_k = 1$, we have $\Delta_{k,2} = 0$. Here $q = -2$ is the only possibility for $h_{k,2}(q) = 0$. Then for any $q \in \mathbb{R}^+$ we have $h_{k,2}(q) > 0$.

If $1 < \mathscr{R}_k < (1+\sqrt{5})/2$, we have $\Delta_{k,2} < 0$. Then the upward parabola $h_{k,2}(\cdot)$ is above zero for any $q \in \mathbb{R}^+$.

If $\mathscr{R}_k = (1+\sqrt{5})/2$, we have $\Delta_{k,2} = 0$. Again, $h_{k,2}(q) > 0$ for any $q \in \mathbb{R}^+$.

If $\mathscr{R}_k > (1+\sqrt{5})/2$, we have $\Delta_{k,2} > 0$. The two roots of $h_{k,2}(\cdot)$ are $2\left(\mathscr{R}_k{}^2 - \mathscr{R}_k - 1\right) - 2\sqrt{q_k(\mathscr{R}_k - 1)\left(\mathscr{R}_k{}^2 - \mathscr{R}_k - 1\right)}$ and $2\left(\mathscr{R}_k{}^2 - \mathscr{R}_k - 1\right) + 2\sqrt{q_k(\mathscr{R}_k - 1)\left(\mathscr{R}_k{}^2 - \mathscr{R}_k - 1\right)} \equiv q_{k,2}$. Since the sum of the two real roots is $4(\mathscr{R}_k^2 - \mathscr{R}_k - 1) > 0$, and the product of the two real roots is $-4(\mathscr{R}_k^2 - \mathscr{R}_k - 1) < 0$, we know that the smaller root is negative and the larger root $q_{k,2}$ is positive. Therefore, $h_{k,2}(\cdot)$ is positive for any q satisfying (3.16). □

It is straightforward to verify that $q_{k,1} > q_{k,2}$ holds[4] for any $\mathscr{R}_k > (1+\sqrt{5})/2$.

[4]The relationship $q_{k,1} > q_{k,2}$ on $\mathscr{R}_k > (1+\sqrt{5})/2$ follows from $d(q_{k,1} - q_{k,2})/d\mathscr{R}_k > 0$ and that $q_{k,1} - q_{k,2}$ evaluated at $\mathscr{R}_k = (1+\sqrt{5})/2$ is approximately $7.35 > 0$.

Lemma 3.3.6 (Slack Variable q_k Selection)**.** *Let us select q_k in the following manner:*

$$q_k \in \mathbb{I}_{\{\mathscr{R}_k=1\}} \times (0,\infty) + \mathbb{I}_{\{1<\mathscr{R}_k\leq(1+\sqrt{5})/2\}} \times (0, q_{k,1}] + \mathbb{I}_{\{\mathscr{R}_k>(1+\sqrt{5})/2\}} \times (q_{k,2}, q_{k,1}]\,, \quad (3.17)$$

where $q_{k,1}$ and $q_{k,2}$ are defined in (3.15) and (3.16) respectively. Specifically, when $\mathscr{R}_k = 1$, let q_k be any number in $\mathbb{R}^+$; when $1 < \mathscr{R}_k \leq (1+\sqrt{5})/2$, let q_k be any number in $(0, q_{k,1}(\mathscr{R}_k)]$; when $\mathscr{R}_k > (1+\sqrt{5})/2$, let q_k be any number in $(q_{k,2}(\mathscr{R}_k), q_{k,1}(\mathscr{R}_k)]$. After selecting the slack variable q_k from the domain corresponding to different values of $\mathscr{R}_k$, the non-decaying gain a_k will be selected such that:

$$\begin{aligned} a_k\mathscr{L}_k \in \mathbb{I}_{\{\mathscr{R}_k=1\}} \times \left[1, 1+\tfrac{1}{q_k+1}\right) + \mathbb{I}_{\{1<\mathscr{R}_k\leq(1+\sqrt{5})/2\}} \times \left[\tfrac{1}{q_k+1}, m_{k,+}(q_k)\right) \\ + \mathbb{I}_{\{\mathscr{R}_k>(1+\sqrt{5})/2\}} \times (m_{k,-}(q_k), m_{k,+}(q_k)) \end{aligned} \quad (3.18)$$

where the mappings $m_{k,\pm}(\cdot)$ are defined as:

$$m_{k,\pm}(q) \equiv \frac{q+2 \pm \sqrt{q^2 + 4(1+\mathscr{R}_k-\mathscr{R}_k^2)q + 4(1+\mathscr{R}_k-\mathscr{R}_k^2)}}{2(q+1)}. \quad (3.19)$$

Specifically, when $\mathscr{R}_k = 1$, let a_k be such that $(q_k+1)^{-1} \leq a_k\mathscr{L}_k < 1 + (q_k+1)^{-1}$; when $1 < \mathscr{R}_k \leq (1+\sqrt{5})/2$, let a_k be such that $(q_k+1)^{-1} \leq a_k\mathscr{L}_k < m_{k,+}(q_k)$; when $\mathscr{R}_k > (1+\sqrt{5})/2$, let a_k be such that $m_{k,-}(q_k) < a_k\mathscr{L}_k < m_{k,+}(q_k)$.

If the slack variable q_k is selected according to (3.17) and then the gain sequence a_k is selected according to (3.18), the following hold:

$$\begin{cases} u_k \equiv \dfrac{\mathscr{L}_k}{\mathscr{C}_k} + a_k \mathscr{C}_k \left[(q_k + 1)(a_k \mathscr{L}_k - 1) - 1 \right] \in [0, 1) \,, & (3.20) \\ v_k \equiv \dfrac{a_k [a_k \mathscr{L}_k \, (q_k + 1) - 1]}{q_k \mathscr{C}_k} \geq 0. & (3.21) \end{cases}$$

Proof of Lemma 3.3.6. First show that $u_k \geq 0$ in (3.20) holds for any a_k satisfying (3.18). We need to show that

$$\tilde{h}_{k,1}(a_k; q_k) \equiv \mathscr{C}_k^2 \mathscr{L}_k \, (q_k + 1) \, a_k^2 - \mathscr{C}_k^2 \, (q_k + 2) \, a_k + \mathscr{L}_k \geq 0 \tag{3.22}$$

holds for all k. The semicolon in (3.22) is to emphasize that the selection of q_k takes place before the selection of a_k. After fixing/picking the value of q_k, $\tilde{h}_{k,1}$ in (3.22) is simply a function of a_k.

When $\mathscr{R}_k = 1$, we pick $q_k > 0$ per (3.17). Then $\tilde{h}_{k,1}(a_k; q_k) \geq 0$ for any a_k such that $0 < a_k \mathscr{L}_k \leq (q_k + 1)^{-1}$ or $a_k \mathscr{L}_k \geq 1$. When $\mathscr{R}_k > 1$, we pick q_k from $(0, q_{k,1}(\mathscr{R}_k)]$ per (3.17). From Lemma 3.3.4 that $\tilde{h}_{k,1}(\cdot)$ has a nonpositive determinant as long as (3.15) holds. In this case, the upward parabola $\tilde{h}_{k,1}(a_k) \geq 0$ for any $a_k \in \mathbb{R}^+$. In short, (3.18) is a sufficient (but not necessary) condition for $u_k \geq 0$.

Next, we show that $u_k < 1$ in (3.20) holds for any a_k satisfying (3.18). We need to show:

$$\tilde{h}_{k,2}(a_k; q_k) \equiv \mathscr{C}_k^2 \mathscr{L}_k (q_k + 1) a_k^2 - \mathscr{C}_k^2 (q_k + 2) a_k + (\mathscr{L}_k - \mathscr{C}_k) < 0. \quad (3.23)$$

When $\mathscr{R}_k = 1$, we pick $q_k > 0$ per (3.17). Then $\tilde{h}_{k,2}(a_k) < 0$ when $0 < a_k \mathscr{L}_k < (q_k + 2)(q_k + 1)^{-1}$.

When $1 < \mathscr{R}_k \leq (1 + \sqrt{5})/2$, we pick $q_k \in (0, q_{k,1}(\mathscr{R}_k)]$ per (3.17). Lemma 3.3.5 tells that $(q_k + 2)^2 - 4\mathscr{R}_k (\mathscr{R}_k - 1)(q_k + 1) > 0$ for any $q_k > 0$. Therefore, we have

$$\tilde{h}_{k,2}(a_k) < 0, \quad \text{when } 0 \leq \frac{m_{k,-}}{\mathscr{L}_k} < a_k < \frac{m_{k,+}}{\mathscr{L}_k} < \infty. \quad (3.24)$$

where $m_{k,\pm}(\cdot)$ are defined in (3.19). For the time being, suppress the parameter q_k in the mappings $m_{k,\pm}(\cdot)$. Notice that both $m_{k,-} + m_{k,+} = (q_k + 2)/(q_k + 1)$ and $m_{k,-} m_{k,+} = \mathscr{R}_k (\mathscr{R}_k - 1)/(q_k + 1)$ are strictly positive except when $\mathscr{R}_k = 1$. Hence, both $m_{k,-}$ and $m_{k,+}$ are strictly positive, except that $m_- = 0$ when $\mathscr{R}_k = 1$.

When $\mathscr{R}_k \geq (1 + \sqrt{5})/2$, we pick $q_k \in (q_{k,2}(\mathscr{R}_k), q_{k,1}(\mathscr{R}_k)]$ per (3.17). Lemma 3.3.5 tells that $(q_k + 2)^2 - 4\mathscr{R}_k (\mathscr{R}_k - 1)(q_k + 1) > 0$ for any q_k satisfying (3.16). Again, we have (3.24).

In short, we have $u_k < 1$ for any a_k satisfying (3.18).

Last, $v_k > 0$ is immediate. This follows from $a_k \mathscr{L}_k \geq (q_k + 1)^{-1}$.

Note that $m_{k,\pm}$ is well-defined for any $\mathscr{R}_k \geq 1$ and the corresponding selection of q_k in (3.17). Given that:

$$0 \leq m_{k,-}(q_k) < \frac{q_k/2+1}{q_k+1} < m_{k,+}(q_k) \leq \frac{q_k+2}{q_k+1}, \tag{3.25}$$

where the first and the last inequalities in (3.25) become strict equalities only when $\mathscr{R}_k = 1$, we can combine the aforementioned scenarios for gain-selection into a consistent expression in (3.18), such that (3.20) and (3.21) always hold for proper slack variable selection (3.17) and gain selection (3.18). □

Lemma 3.3.6 discusses both the slack variable q_k and the gain sequence a_k. To facilitate reading and implementation, we summarize the implementation of (2.1) as applied in tracking time variability in Algorithm 1.

The motivation behind the restriction on the non-diminishing gain a_k in Lemma 3.3.6 is to ensure the tracking capability, with a manifestation of the shrinking recurrence coefficient in the inequality (3.26) below.

Lemma 3.3.7 (Error propagation)**.** *Assume that A.1, A.2, A.3, and A.4 hold. Let q_k be a slack variable selected according to (3.17), the domain of which depends on different values of $\mathscr{R}_k$. Let $\hat{\theta}_k$ be the sequence generated by (2.1) with non-decaying gain satisfying (3.18). We have:*

$$\|\hat{\theta}_{k+1} - \theta_k^*\|^2 \leq u_k\|\hat{\theta}_k - \theta_k^*\|^2 + v_k\|\boldsymbol{e}_k(\hat{\theta}_k)\|^2, \tag{3.26}$$

Algorithm 1 Basic SA Algorithm With Non-Diminishing Gain Using *One*-Function-Measurement or *One*-Stochastic-Gradient-Measurement Per Iteration

Input: $\hat{\theta}_0$, the best approximation available at hand to estimate θ_0^*.

1: **for** $k \geq 0$ or $k \in \{0, 1, \cdots, K\}$ **do**

Input: $\mathscr{C}_k$ per A.2 and $\mathscr{L}_k$ per A.3.

2: **collect** instant feedback immediately after each decision.

3: **if** the feedback is in the form of (3.2) **then**

4: **generate** a p-dimensional random vector $\boldsymbol{\Delta}_k$ satisfying conditions in [106, Sect. 7.3]. ▷ Each component of $\boldsymbol{\Delta}_k$ may be i.i.d. Rademacher distributed.

5: **pick** a small number c_k. ▷ c_k may be the desired minimal component-wise change of $\hat{\theta}_k$.

6: **compute** $\hat{\boldsymbol{g}}_k^{\mathrm{SP1}}(\hat{\theta}_k)$ using (3.4).

7: **else if** the feedback is in the form of (3.3) **then**

8: **set** $\hat{\boldsymbol{g}}_k^{\mathrm{RM}}(\hat{\theta}_k)$ as in (2.5).

9: **end if**

10: **if** $\mathscr{R}_k = 1$ **then**

11: **pick** any $q_k \in \mathbb{R}^+$.

12: **pick** a_k such that $1 \leq a_k \mathscr{L}_k < 1 + (q_k + 1)^{-1}$.

13: **else if** $1 < \mathscr{R}_k \leq (1 + \sqrt{5})/2$ **then**

14: **pick** any $q_k \in (0, q_{k,1}(\mathscr{R}_k)]$ per (3.15).

15: **pick** a_k such that $a_k \mathscr{L}_k \in [(q_k + 1)^{-1}, m_{k,+}(q_k))$ per (3.19).

16: **else if** $\mathscr{R}_k > (1 + \sqrt{5})/2$ **then**

17: **pick** any $q \in (q_{k,2}(\mathscr{R}_k), q_{k,1}(\mathscr{R}_k)]$ per (3.15) and (3.16).

18: **pick** a_k such that $a_k \mathscr{L}_k \in (m_{k,-}(q_k), m_{k,+}(q_k))$ per (3.19).

19: **end if**

20: **update** $\hat{\theta}_k$ using (2.1).

Output: $\hat{\theta}_k$.

21: **end for**

where the constants u_k and v_k were defined in (3.20) and (3.21). Both coefficients are deterministic and can be fully determined after selecting q_k and a_k per Algorithm 1. Furthermore, the coefficient u_k is guaranteed to lie within $[0, 1)$, and the coefficient v_k is positive.

Proof of Lemma 3.3.7. The estimates generated by (2.1) satisfy the following:

$$\begin{aligned}
f_k(\hat{\theta}_{k+1}) &= f_k(\hat{\theta}_k - a_k \hat{\boldsymbol{g}}_k(\hat{\theta}_k)) \\
&\leq f_k(\hat{\theta}_k) - a_k[\hat{\boldsymbol{g}}_k(\hat{\theta}_k)]^T \boldsymbol{g}_k(\hat{\theta}_k) + \frac{a_k^2 \mathscr{L}_k}{2} \|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|^2 \\
&= f_k(\hat{\theta}_k) - a_k[\boldsymbol{g}_k(\hat{\theta}_k) + \boldsymbol{e}_k(\hat{\theta}_k)]^T \boldsymbol{g}_k(\hat{\theta}_k) + \frac{a_k^2 \mathscr{L}_k}{2} \|\boldsymbol{g}_k(\hat{\theta}_k) + \boldsymbol{e}_k(\hat{\theta}_k)\|^2 \\
&= f_k(\hat{\theta}_k) + a_k \left(a_k \mathscr{L}_k/2 - 1\right) \|\boldsymbol{g}_k(\hat{\theta}_k)\|^2 + a_k(a_k \mathscr{L}_k - 1)[\boldsymbol{g}_k(\hat{\theta}_k)]^T \boldsymbol{e}_k(\hat{\theta}_k) \\
&\quad + \frac{a_k^2 \mathscr{L}_k}{2} \|\boldsymbol{e}_k(\hat{\theta}_k)\|^2 \\
&\leq f_k(\hat{\theta}_k) + a_k \left(\frac{a_k \mathscr{L}_k}{2} - 1\right) \|\boldsymbol{g}_k(\hat{\theta}_k)\|^2 + \frac{a_k^2 \mathscr{L}_k}{2} \|\boldsymbol{e}_k(\hat{\theta}_k)\|^2 \\
&\quad \frac{a_k(a_k \mathscr{L}_k - 1)}{2} \left[q_k \|\boldsymbol{g}_k(\hat{\theta}_k)\|^2 + q_k^{-1} \|\boldsymbol{e}_k(\hat{\theta}_k)\|^2\right] \\
&= f_k(\hat{\theta}_k) + \frac{a_k}{2} \left[(a_k \mathscr{L}_k - 1)(1 + q_k) - 1\right] \|\boldsymbol{g}_k(\hat{\theta}_k)\|^2 \\
&\quad + \frac{a_k}{2q_k} [a_k \mathscr{L}_k (1 + q_k) - 1] \|\boldsymbol{e}_k(\hat{\theta}_k)\|^2, \qquad (3.27)
\end{aligned}$$

where the first inequality follows from (3.12), the second equality follows from the fact that $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ on the r.h.s. can be decomposed as (3.5), and the second inequality follows from the fact that $\pm 2\boldsymbol{x}^T \boldsymbol{y} \leq \|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2$ for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^p$. Note that both $2\boldsymbol{x}^T \boldsymbol{y} \leq \|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2$ and $-2\boldsymbol{x}^T \boldsymbol{y} \leq \|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2$ hold, so the sign of the coefficient $a_k(a_k \mathscr{L}_k - 1)/2$ does not

affect the validity of the inequality. The positive slack variable q_k, which may be deemed as hyper-parameter for selecting selection a_k, is picked according to Lemma 3.3.6 and the corresponding gain selection is summarized in Algorithm 1.

Subtracting $f_k(\theta_k^*)$ from both sides of (3.27) yields:

$$f_k(\hat{\theta}_{k+1}) - f_k(\theta_k^*) \leq f_k(\hat{\theta}_k) - f_k(\theta_k^*) + \frac{a_k}{2}\left[(a_k\mathscr{L}_k - 1)(1+q_k) - 1\right]\|\boldsymbol{g}_k(\hat{\theta}_k)\|^2 + \frac{a_k}{2q_k}[a_k\mathscr{L}_k(1+q_k) - 1]\|\boldsymbol{e}_k(\hat{\theta}_k)\|^2. \tag{3.28}$$

Applying both inequalities in (3.7) to both sides of (3.28) gives:

$$\begin{aligned}
\mathscr{C}_k\|\hat{\theta}_{k+1} - \theta_k^*\|^2/2 &\leq f_k(\hat{\theta}_{k+1}) - f_k(\theta_k^*) \\
&\leq f_k(\hat{\theta}_k) - f_k(\theta_k^*) + \frac{a_k}{2}\left[(a_k\mathscr{L}_k - 1)(1+q_k) - 1\right]\|\boldsymbol{g}_k(\hat{\theta}_k)\|^2 \\
&\quad + \frac{a_k}{2q_k}[a_k\mathscr{L}_k(1+q_k) - 1]\|\boldsymbol{e}_k(\hat{\theta}_k)\|^2 \\
&\leq \mathscr{L}_k\|\hat{\theta}_k - \theta_k^*\|^2/2 + \frac{a_k}{2}\left[(a_k\mathscr{L}_k - 1)(1+q_k) - 1\right]\|\boldsymbol{g}_k(\hat{\theta}_k)\|^2 \\
&\quad + \frac{a_k}{2q_k}[a_k\mathscr{L}_k(1+q_k) - 1]\|\boldsymbol{e}_k(\hat{\theta}_k)\|^2,
\end{aligned}$$

which implies

$$\|\hat{\theta}_{k+1} - \theta_k^*\|^2 \leq \frac{\mathscr{L}_k}{\mathscr{C}_k}\|\hat{\theta}_k - \theta_k^*\|^2 + \frac{a_k}{\mathscr{C}_k}\left[(q_k+1)(a_k\mathscr{L}_k - 1) - 1\right]\|\boldsymbol{g}_k(\hat{\theta}_k)\|^2 + \frac{a_k}{q_k\mathscr{C}_k}[a_k\mathscr{L}_k\,(q_k+1) - 1]\|\boldsymbol{e}_k(\hat{\theta}_k)\|^2. \tag{3.29}$$

(3.25) tells that $a_k{\mathscr{C}_k}^{-1}\left[(q_k+1)(a_k\mathscr{L}_k-1)-1\right]$, the coefficient of $\|\boldsymbol{g}_k(\hat{\boldsymbol{\theta}}_k)\|^2$ on the r.h.s. of (3.29), is negative as long as both the positive slack variable q_k and the non-diminishing step-size a_k are selected per (3.17) and (3.18) in Lemma 3.3.6. Hence, we can apply (3.7) and (3.8) to further manipulate (3.29):

$$\begin{aligned}
\|\hat{\boldsymbol{\theta}}_{k+1}-\boldsymbol{\theta}_k^*\|^2 &\leq \frac{\mathscr{L}_k}{\mathscr{C}_k}\|\hat{\boldsymbol{\theta}}_k-\boldsymbol{\theta}_k^*\|^2+a_k\mathscr{C}_k\left[(q_k+1)(a_k\mathscr{L}_k-1)-1\right]\|\hat{\boldsymbol{\theta}}_k-\boldsymbol{\theta}_k^*\|^2 \\
&\quad +\frac{a_k}{q_k\mathscr{C}_k}[a_k\mathscr{L}_k\left(q_k+1\right)-1]\|\boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_k)\|^2 \\
&= \frac{\mathscr{L}_k+a_k\mathscr{C}_k^2\left[\left(q_k+1\right)\left(a_k\mathscr{L}_k-1\right)-1\right]}{\mathscr{C}_k}\|\hat{\boldsymbol{\theta}}_k-\boldsymbol{\theta}_k^*\|^2 \\
&\quad +\frac{a_k[a_k\mathscr{L}_k\left(q_k+1\right)-1]}{q_k\mathscr{C}_k}\|\boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_k)\|^2 \\
&\equiv u_k\|\hat{\boldsymbol{\theta}}_k-\boldsymbol{\theta}_k^*\|^2+v_k\|\boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_k)\|^2. \qquad (3.30)
\end{aligned}$$

Note that both u_k and v_k are deterministic once a_k is selected following Lemma 3.3.6. Algorithm 1 summarizes the gain selection procedure. Furthermore, the coefficient u_k on the r.h.s. of (3.30) is guaranteed to lie within $[0,1)$ as long as (3.17) and (3.18) hold per Lemma 3.3.6. The coefficient v_k on the r.h.s. of (3.30) is guaranteed to be positive per Lemma 3.3.6. □

Admittedly, Lemma 3.3.7 is presented in a way that the slack variable q_k also appears in the required conditions. To present our theorem with conditions imposed solely on the gain sequence a_k, the only hyper-parameter in (2.1), Lemma 3.3.8 is presented below.

Lemma 3.3.8. *Let us use the recursion (2.1) where the non-diminishing gain is such that $a_k\mathscr{L}_k$ lies within the shaded (green) region (excluding the red boundaries) in Figure 3.1.*

The region in Figure 3.1 is confined by the lower curve defined to be

$$\mathbb{I}_{\{1<\mathscr{R}_k\leq(1+\sqrt{5})/2\}}\times\frac{1}{1+q_{k,1}(\mathscr{R}_k)}+\mathbb{I}_{\{\mathscr{R}_k>(1+\sqrt{5})/2\}}\times m_{k,-}(q_{k,1}(\mathscr{R}_k))$$

as a function of $\mathscr{R}_k$, and the upper curve defined to be

$$\mathbb{I}_{\{1<\mathscr{R}_k\leq(1+\sqrt{5})/2\}}\times m_{k,+}(0)+\mathbb{I}_{\{\mathscr{R}_k>(1+\sqrt{5})/2\}}\times m_{k,+}(q_{k,1}(\mathscr{R}_k))$$

as a function of $\mathscr{R}_k$. When Assumptions A.2 and A.3 hold, and a_k falls within the green

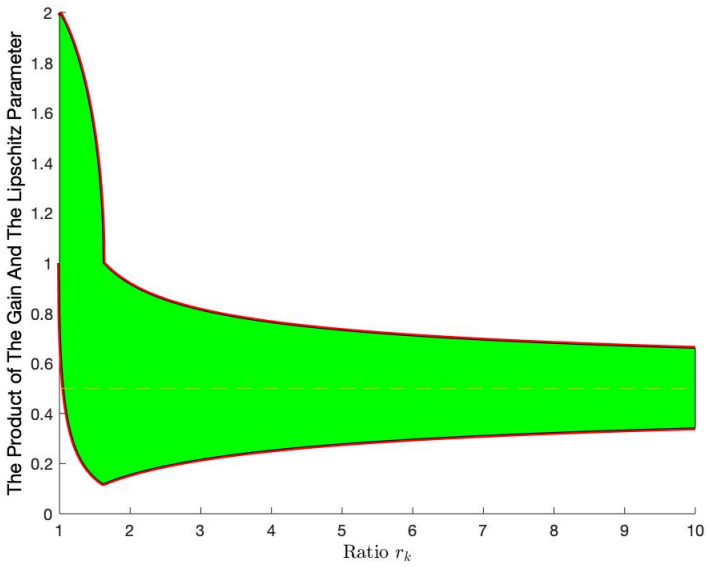


Figure 3.1: The Allowable Region For The Product of The Gain and The Lipschitz Parameter $a_k\mathscr{L}_k$ Under Different Values of The Ratio $\mathscr{R}_k$.

region indicated in Figure 3.1, we have that (3.26) holds, where u_k is guaranteed to lie within $[0,1)$ and v_k is nonnegative.

Proof of Lemma 3.3.8. For $q_{k,1}(\mathscr{R}_k)$ defined in (3.15), we know that $q_{k,1}(\mathscr{R}_k)$ is strictly increasing w.r.t. $\mathscr{R}_k$ as $\partial q_{k,1}/\partial \mathscr{R}_k$ is strictly positive on $\mathscr{R}_k > 1$. Also, $q_{k,2}$ defined in (3.16) is also a strictly increasing function of $\mathscr{R}_k$ because $\partial q_{k,2}/\partial \mathscr{R}_k$ is strictly positive on $\mathscr{R}_k > (1+\sqrt{5})/2$.

When $\mathscr{R}_k = 1$, lines 11–12 in Algorithm 1 immediately tell $1 \leq a_k \mathscr{L}_k < 2$.

When $1 < \mathscr{R}_k \leq (1+\sqrt{5})/2$, lines 14 and 15 instruct selecting any a_k such that $(q_k+1)^{-1} \leq a_k \mathscr{L}_k < m_{k,+}(q_k)$ for $0 < q_k \leq q_{k,1}(\mathscr{R}_k)$. In particular, $(1+q_{k,1}(\mathscr{R}_k))^{-1} < a_k \mathscr{L}_k < m_{k,+}(0)$, where $m_{k,+}(0) = 1 + \sqrt{1+\mathscr{R}_k - \mathscr{R}_k^2}$.

When $\mathscr{R}_k > (1+\sqrt{5})/2$, lines 17 and 18 allow for selecting any a_k such that $m_{k,-}(q_k) < a_k \mathscr{L}_k < m_{k,+}(q_k)$ for $q_{k,2}(\mathscr{R}_k) < q_k \leq q_{k,1}(\mathscr{R}_k)$. In particular, $m_{k,-}(q_{k,1}(\mathscr{R}_k)) < a_k \mathscr{L}_k < m_{k,+}(q_{k,1}(\mathscr{R}_k))$. □

3.3.2 A Priori Error Bound

For time-varying systems, it is unrealistic to expect the convergence results as in classic SA settings; i.e., for $\|\hat{\theta}_k - \theta_k^*\|$ to be arbitrarily close to zero in a certain statistical sense. Our first main result pertains to the error propagation in terms of MAD (see Theorem 3.3.1).

Theorem 3.3.1 (MAD bound under bounded-drift assumption)**.** *Assume A.1, A.2, A.3, and A.4. Using recursion (2.1) with non-decaying gain satisfying the region specified in Lemma 3.3.8, we have the following recurrence on the unconditional MAD:*

$$\mathbb{E}\|\hat{\theta}_{k+1} - \theta_{k+1}^*\| \leq \sqrt{u_k}\,\mathbb{E}\|\hat{\theta}_k - \theta_k^*\| + \mathscr{M}_k\sqrt{v_k} + \mathscr{B}_k, \quad k \in \mathbb{N}. \tag{3.31}$$

Furthermore, we have an asymptotic bound

$$\limsup_{k\to\infty} \mathbb{E}\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\| \leq \limsup_{k} \frac{\mathscr{M}_k\sqrt{v_k} + \mathscr{B}_k}{1-\sqrt{u_k}}. \tag{3.32}$$

Proof of Theorem 3.3.1. Note that $\sqrt{x^2+y^2} < x + y$ for $x, y \in \mathbb{R}^+$, then from (3.26) we have the following:

$$\|\hat{\boldsymbol{\theta}}_{k+1} - \boldsymbol{\theta}_k^*\| \leq \sqrt{u_k}\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\| + \sqrt{v_k}\|\boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_k)\|, \tag{3.33}$$

where $u_k \in [0, 1)$ and $v_k \geq 0$. Note that both u_k and v_k in Lemma 3.3.7, though flexible, are deterministic. By Jensen's inequality, A.1 implies that $\mathbb{E}\|\boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_k)\| \leq \mathscr{M}_k$ for all $\boldsymbol{\theta} \in \mathbb{R}^p$ and $k \in \mathbb{Z}^+$. Similarly, A.4 implies that $\mathbb{E}\|\boldsymbol{\theta}_{k+1}^* - \boldsymbol{\theta}_k^*\| \leq \mathscr{B}_k$ for all k. Taking the full expectation over (3.33) and invoking A.1, we have:

$$\mathbb{E}\|\hat{\boldsymbol{\theta}}_{k+1} - \boldsymbol{\theta}_k^*\| \leq \sqrt{u_k}\mathbb{E}\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\| + \mathscr{M}_k\sqrt{v_k}.$$

Then (3.31) directly follows from triangle inequality and A.4 as:

$$\begin{aligned}
\mathbb{E}\|\hat{\boldsymbol{\theta}}_{k+1} - \boldsymbol{\theta}_{k+1}^*\| &\leq \mathbb{E}\|\hat{\boldsymbol{\theta}}_{k+1} - \boldsymbol{\theta}_k^*\| + \mathbb{E}\|\boldsymbol{\theta}_{k+1}^* - \boldsymbol{\theta}_k^*\| \\
&\leq \sqrt{u_k}\mathbb{E}\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\| + \mathscr{M}_k\sqrt{v_k} + \mathscr{B}_k.
\end{aligned} \tag{3.34}$$

Define $\nu_{j,k} \equiv (1-\sqrt{u_j})\prod_{i=j+1}^{k}\sqrt{u_i}$ for $j < k$. Then after iterating inequality (3.34) back to the starting time index 0, we have

$$\mathbb{E}\|\hat{\theta}_k - \theta_k^*\| \leq \left(\prod_{j=0}^{k-1}\sqrt{u_i}\right)\mathbb{E}\|\hat{\theta}_0 - \theta_0^*\| + \sum_{j=0}^{k-1}\frac{\nu_{j,k-1}\left(\mathscr{M}_j\sqrt{v_j}+\mathscr{B}_j\right)}{1-\sqrt{u_j}}, \quad k \geq 1. \tag{3.35}$$

Since the u_k are bounded within $[0,1)$, the leading product $\prod_{j=0}^{k-1}\sqrt{u_i}$ goes to zero as $k \to \infty$. According to Lemma 3.3.2, we know that $\sum_{j=0}^{k-1}\nu_{j,k-1} = 1 - \prod_{j=0}^{k-1}\sqrt{u_j}$, which goes to 1 as $k \to \infty$. Now we can apply Lemma 3.3.3 to conclude that the asymptotic bound (3.32) holds. □

To present the theorem in terms of RMS, we need the following lemma.

Lemma 3.3.9 (Triangle Inequality for RMS)**.** *For any p-dimensional random vectors $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$, we have:*

$$\sqrt{\mathbb{E}\left(\|\boldsymbol{x}-\boldsymbol{y}\|^2\right)} \leq \sqrt{\mathbb{E}\left(\|\boldsymbol{x}-\boldsymbol{z}\|^2\right)} + \sqrt{\mathbb{E}\left(\|\boldsymbol{z}-\boldsymbol{y}\|^2\right)}.$$

Proof of Lemma 3.3.9. We will show that the r.h.s. squared is greater than or equal to the LHS squared.

$$\begin{aligned}
&\mathbb{E}(\|\boldsymbol{x}-\boldsymbol{z}\|^2)+\mathbb{E}(\|\boldsymbol{z}-\boldsymbol{y}\|^2)+2\sqrt{[\mathbb{E}(\|\boldsymbol{x}-\boldsymbol{z}\|^2)][\mathbb{E}(\|\boldsymbol{z}-\boldsymbol{y}\|^2)]}\\
&\quad\geq \mathbb{E}(\|\boldsymbol{x}-\boldsymbol{z}\|^2)+\mathbb{E}(\|\boldsymbol{z}-\boldsymbol{y}\|^2)+2\mathbb{E}(\|\boldsymbol{x}-\boldsymbol{z}\|\|\boldsymbol{z}-\boldsymbol{y}\|)\\
&\quad= \mathbb{E}\left[(\|\boldsymbol{x}-\boldsymbol{z}\|+\|\boldsymbol{z}-\boldsymbol{y}\|)^2\right]\\
&\quad\geq \mathbb{E}(\|\boldsymbol{x}-\boldsymbol{y}\|^2),
\end{aligned}$$

where the first inequality follows from the Cauchy-Schwartz inequality, and the last inequality follows from the triangle inequality. □

Theorem 3.3.2 (RMS bound under bounded-drift assumption)**.** *Assume A.1, A.2, A.3, and A.4. Using recursion (2.1) with non-decaying gain satisfying the region specified in Lemma 3.3.8, we have the following recurrence on the unconditional RMS:*

$$\sqrt{\mathbb{E}(\|\hat{\boldsymbol{\theta}}_{k+1}-\boldsymbol{\theta}^*_{k+1}\|^2)}\leq\sqrt{u_k\mathbb{E}(\|\hat{\boldsymbol{\theta}}_k-\boldsymbol{\theta}^*_k\|^2)}+\mathscr{M}_k\sqrt{v_k}+\mathscr{B}_k,\quad k\in\mathbb{N}. \tag{3.36}$$

Furthermore, we have an asymptotic bound

$$\limsup_{k\to\infty}\sqrt{\mathbb{E}(\|\hat{\boldsymbol{\theta}}_k-\boldsymbol{\theta}^*_k\|^2)}\leq\limsup_k\frac{\mathscr{M}_k\sqrt{v_k}+\mathscr{B}_k}{1-\sqrt{u_k}}. \tag{3.37}$$

Proof of Theorem 3.3.2. Recall that both u_k and v_k in Lemma 3.3.7 are deterministic. Take the full expectation over (3.26) and invoke A.1:

$$\begin{aligned}\mathbb{E}(\|\hat{\boldsymbol{\theta}}_{k+1} - \boldsymbol{\theta}_k^*\|^2) &\leq u_k \mathbb{E}(\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\|^2) + v_k \mathbb{E}(\|\boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_k)\|^2) \\ &\leq u_k \mathbb{E}(\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\|^2) + v_k \mathscr{M}_k^2, \end{aligned} \tag{3.38}$$

which implies that:

$$\sqrt{\mathbb{E}(\|\hat{\boldsymbol{\theta}}_{k+1} - \boldsymbol{\theta}_k^*\|^2)} \leq \sqrt{u_k}\sqrt{\mathbb{E}(\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\|^2)} + \mathscr{M}_k\sqrt{v_k}, \tag{3.39}$$

because $u_k \in [0, 1)$, $v_k \geq 0$, and that $\sqrt{x^2 + y^2} < x + y$ for $x, y \in \mathbb{R}^+$. Then (3.36) follows directly from Lemma 3.3.9:

$$\begin{aligned}\sqrt{\mathbb{E}(\|\hat{\boldsymbol{\theta}}_{k+1} - \boldsymbol{\theta}_{k+1}^*\|^2)} &\leq \sqrt{\mathbb{E}(\|\hat{\boldsymbol{\theta}}_{k+1} - \boldsymbol{\theta}_k^*\|^2)} + \sqrt{\mathbb{E}(\|\boldsymbol{\theta}_{k+1}^* - \boldsymbol{\theta}_k^*\|^2)} \\ &\leq \sqrt{u_k}\sqrt{\mathbb{E}(\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\|^2)} + \mathscr{M}_k\sqrt{v_k} + \mathscr{B}_k. \end{aligned} \tag{3.40}$$

Now following the derivation immediately after equation (3.34) in the proof for Theorem 3.3.1, we can obtain the asymptotic bound (3.37). □

Consequent Tightness

As mentioned in Section 4.1, the best we can hope for in the time-varying scenario is that the error term $(\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*)$ hovers near zero, and that our concern centers on boundedness

(input-output stability). The notion that no probability mass escapes to infinity uniformly in k is termed as tightness, which was reviewed in Subsection 2.4.2.

$$\lim_{M\to\infty} \sup_{k} \mathbb{P}\left\{\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\| \geq M\right\} = 0. \tag{3.41}$$

By Chebyshev's inequality, $\mathbb{P}\{\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\| \geq M\} \leq \mathbb{E}\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\|^2/M^2$ holds, where the inequality follows from (3.37) in Theorem 3.3.2, the boundedness of both u_k and v_k given the selection of slack variable and gain sequence in Lemma 3.3.6, the assumed finiteness of $\mathscr{M}_k$ in A.4, $\mathscr{C}_k$ in A.2, $\mathscr{L}_k$ in A.3, and $\mathscr{B}_k$ in A.4. Therefore, the boundedness in probability follows, and so does the mean square boundedness.

Further Remarks on Slack Variables and Gain Selection

Let us return to q_k (whose domain depends on $\mathscr{R}_k$) and a_k (whose domain depends on q_k) in Lemma 3.3.6. Recall that they are selected according to (3.17) and (3.18) such that u_k in (3.26) is shrinking. There are, seemingly, many ways we may pursue to optimize the selection of both the slack variable and the gain. For example, we may consider:

i) minimizing u_k defined in (3.20) such that the previous tracking error is "washed away" as quickly as possible,

ii) minimizing the r.h.s. of (3.32) or (3.37) such that the limiting tracking error is as small as possible.

Unfortunately, i) has no attainable minimizer. For ii), the solution depends not only on $\mathscr{R}_k$ but also on $\mathscr{M}_k$ and $\mathscr{B}_k$. Worse still, the quantitative relations between them, in addition to (3.10), also influence the result substantially. Also, note that if our focus is on finite-sample performance, then performing ii) will not benefit us in this sense much after all.

Setting the nonexistence of "optimal" slack variable and gain selections aside, the following observation adds to the difficulty in tuning. A moment of reflection tells us that, even if either problem i) or problem ii) is solvable, the resulting gain may still perform poorly because the derivation in Lemma 3.3.7 only requires a *local* Lipschitz constant, which are usually smaller than the *global* one. Similarly, Lemma 3.3.7 in fact requires a *local* strong convexity parameter, which is usually larger than the *global* one.

In short, we can provide neither an optimal slack variable nor and optimal gain selection. Nonetheless, for ease of implementation, we may set $q_k = q_{k,1}$ and $a_k = 0.5/\mathscr{L}_k$ when $\mathscr{R}_k \geq (1+\sqrt{5})/2$ for simplicity—this is often smaller than what is desired due to the distinction between the global- and local-smoothness parameter. Then, the gain strategy in Lemma 3.3.8 only ensures the parameter u_k in (3.26) is shrinking so that the asymptotic error bounds (3.32) and (3.37) are valid.

The expression on the r.h.s. of (3.32) or (3.37) conveys information for target tracking. The MAD/RMS bound depends explicitly on the noise magnitude $\mathscr{M}_k$ in A.1 and the drift magnitude $\mathscr{B}_k$ in A.4. Besides, it implicitly depends on $\mathscr{C}_k$ in A.2, $\mathscr{L}_k$ in A.3 and the gain a_k selected by the agent(s) through both u and v. The first two dependencies are easy to

understand, as we do expect the bound to be larger when either $\mathscr{M}_k$ or $\mathscr{B}_k$ gets larger. The appearances of $\mathscr{C}_k$ and $\mathscr{L}_k$ in the third dependency are reasonable, as the shape of $\{f_k\}$ does impact our tracking accuracy. Interestingly, both u_k in (3.20) and v_k in (3.21) being *quadratic* functions of a_k inform that the gain a_k for successful tracking should be "neither too large nor too small."

Take u_k as an example and consider $\mathscr{R}_k > 1$ for all k. We pick $q_k = q_{k,1}$ for all k. Then u_k approaches the upper-bound 1 when $a_k\mathscr{L}_k \to m_{k,-}(q_{k,1})$ from the right or $a_k\mathscr{L}_k \to m_{k,+}(q_{k,1})$ from the left. Additionally, u_k approaches the lower-bound 0 when $a_k\mathscr{L}_k = (q_{k,1}/2 + 1)(q_{k,1} + 1)^{-1}$, which is the midpoint of $m_{k,-}(q_{k,1})$ and $m_{k,+}(q_{k,1})$. Similarly, we can discuss v and potentially the coefficient $\sqrt{v_k}/(1 - \sqrt{u_k})$. However, as we do not have a definite objective towards which the slack variable and the gain selection are optimized, we no longer dwell on this topic here.

One Quick Example

This example is borrowed from the illustration for adaptive control in [106, Example 4.7]. Target tracking is a common specific case of control problems. We want to track the coordinates of a time-varying multi-dimensional target, when only the noisy measurements of the distance to the moving target are available. To minimize the distance between the estimate $\hat{\theta}_k$ and the time-varying parameter sequence θ_k^*, we can formulate the loss function as $f_k(\theta) = (\theta - \theta_k^*)^T \boldsymbol{H}(\theta - \theta_k^*)/2$. The true gradient sequence is $\boldsymbol{g}_k(\theta) = \boldsymbol{H}(\theta - \theta_k^*)$, and the true Hessian sequence is $\boldsymbol{H}$.

Consider a simple case with $p = 2$. We construct $\boldsymbol{H} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^T$, where $\boldsymbol{P}$ is (randomly generated) orthogonal and $\boldsymbol{D}$ is diagonal. In our simulation,

$$\boldsymbol{D} = \begin{pmatrix} 30 & 0 \\ 0 & 5 \end{pmatrix}, \quad \boldsymbol{P} = \begin{pmatrix} 0.8145 & -0.5802 \\ -0.5802 & -0.8145 \end{pmatrix}, \quad \text{and} \quad \boldsymbol{H} = \begin{pmatrix} 14.9505 & -7.0884 \\ -7.0884 & 10.0495 \end{pmatrix}. \tag{3.42}$$

The accessible information is the noisy gradient measurement $\boldsymbol{Y}_k(\theta) = \boldsymbol{g}_k(\theta) + \boldsymbol{e}_k$, where $\boldsymbol{e}_k \overset{\text{i.i.d.}}{\sim} \text{Normal}(\boldsymbol{0}, \sigma_1^2 \boldsymbol{I}_p)$. The (unknown to the algorithm) nonstationary drift evolves according to:

$$\theta_{k+1}^* = \begin{cases} \theta_k^* + (1,\ 1)^T + \boldsymbol{w}_k, & \text{for } 1 \le k \le 499, \\ \theta_k^* + (-1,\ 1)^T + \boldsymbol{w}_k, & \text{for } 500 \le k \le 999, \end{cases} \tag{3.43}$$

with $\theta_0^* = \boldsymbol{0}$ and $\boldsymbol{w}_k \overset{\text{i.i.d.}}{\sim} \text{Normal}(\boldsymbol{0}, \sigma_2^2 \boldsymbol{I}_p)$. From Theorem 3.3.1, we take $\mathscr{M}_k = \sqrt{2}\sigma_1$, $\mathscr{B}_k = \sqrt{2}$, and $\mathscr{C}_k = 5$, $\mathscr{L}_k = 30$ for all k. In this experiment, both σ_1 and σ_2 are set to be 10. Note that 10 is large compared to the magnitude of the deterministic trend in $\{\theta_k^*\}$; i.e., heading northeast with step $(1,\ 1)^T$ for $0 \le k \le 499$ and heading northwest with step $(-1,\ 1)^T$ for $500 \le k \le 999$. Both θ_0^* and $\hat{\theta}_0$ are set to $\boldsymbol{0}$. We implement (2.1) for 25 trial runs, each with 1000 iterations.

Following Algorithm 1, we pick $q_k = 0.4q_{k,1} + 0.6q_{k,2}$ and $a = 0.5/\mathscr{L}_k$. Figure 3.2 displays how the iterates generated by (2.1) keep up with the moving target. Figures 3.3a and 3.3b show the accuracy of (3.32) and (3.37) in bounding the tracking error

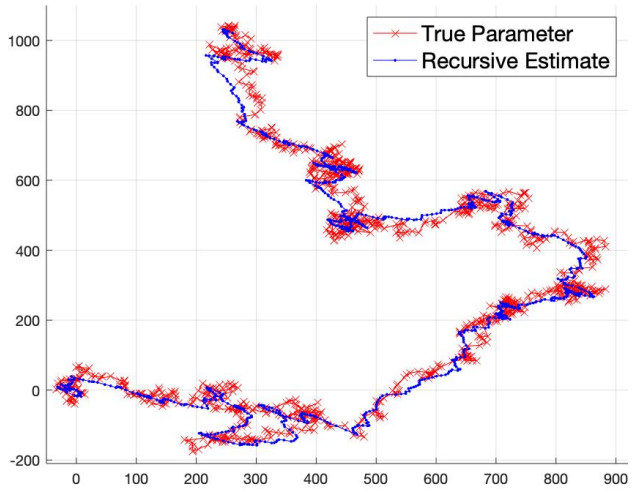
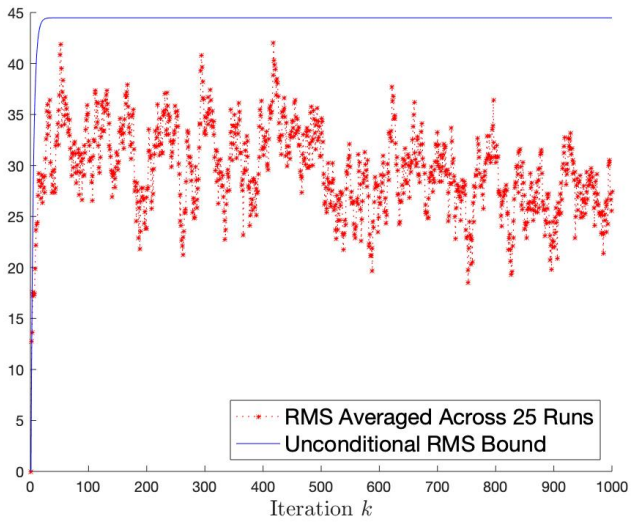


Figure 3.2: The Trajectories Of The True Parameter θ_k^* And The Recursive Estimates $\hat{\theta}_k$ In *One* Run

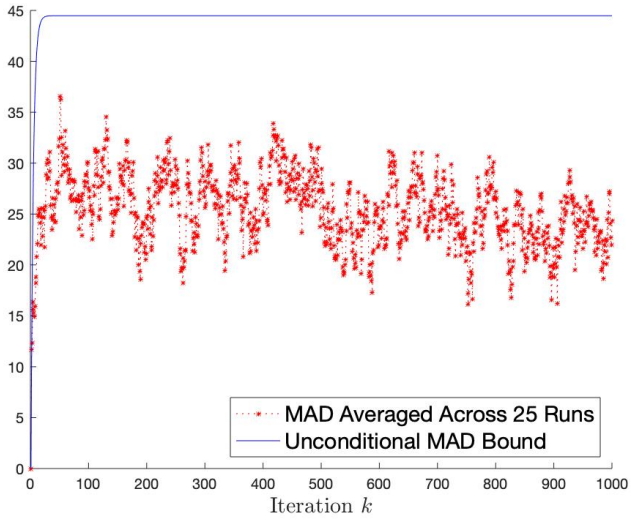
in nonstationary optimization. The empirical MAD/RMS is computed by averaging the absolute-deviation and by taking the root of the averaged squared-error across 25 trial runs. Note that the bounds (3.32) and (3.37), although conservative, are quite accurate in terms of characterizing the empirical error.

3.3.3 A Posterior Error Bound

Applications to physical systems in Subsection 3.3.2 often encourages the "few measurements at a time" requirement explained in Section 3.1. Furthermore, the physical constraints explain that the drift magnitude bound $\mathscr{B}_k$ introduced in A.4 should be knowable or estimatable (similar to (6.14) for the multi-agent application in Chapter 6) in advance and should also be small (relative to the magnitude of the observable gradient information) in Subsection 3.2.7. Note that (3.32) to (3.37) already average out the



(a) The Empirical RMS Averaged Across 25 Runs and The Upper Bound To The RMS



(b) The Empirical MAD Averaged Across 25 Runs and The Upper Bound To The MAD

Figure 3.3: The Empirical Errors (Averaged Across 25 Runs) And The Corresponding Upper Bounds

observable information $\mathcal{F}_k$ which was defined in (2.3). That is, the error bounds in Subsection 3.3.2 provide the a priori tracking performances that are average over all possible sample paths so as to ensure tracking performance. The gain selection in Algorithm 1 is not impacted by the observable information $\mathcal{F}_k$

Nonetheless, during actual implementation, we hope to react to the changes in $\{\theta_k^*\}$ as promptly as possible. Thus, the average performance may not be informative in *one* realization, though it is meaningful in providing gain selection guidance to ensure tracking. In fact, we have the following upper and lower bounds on the MAD, $\mathbb{E}_{k+1}\|\hat{\theta}_k - \theta_k^*\|$, conditioned on $\mathcal{F}_{k+1}$ defined in (2.3), which is the observable information through time instant k.

Theorem 3.3.3 (Conditional MAD bound)**.** *Assume A.1, A.2, and A.3. We have the following MAD bound conditioned on $\mathcal{F}_{k+1}$:*

$$\mathscr{L}_k^{-1}\left|\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\| - \mathscr{M}_k\right| \leq \mathbb{E}_{k+1}\|\hat{\theta}_k - \theta_k^*\| \leq \mathscr{C}_k^{-1}\left(\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\| + \mathscr{M}_k\right), \tag{3.44}$$

where $\mathbb{E}_{k+1}$ is the expectation conditioned on the observable information $\mathcal{F}_{k+1}$ through time instant k.

Proof of Theorem 3.3.3. From (3.8) and (3.7), we have:

$$\mathscr{L}_k^{-1}\|\boldsymbol{g}_k(\theta)\| \leq \|\theta - \theta_k^*\| \leq \mathscr{C}_k^{-1}\|\boldsymbol{g}_k(\theta)\|, \tag{3.45}$$

for any $\theta \in \mathbb{R}^p$ when A.2 and A.3 hold. By the triangle inequality and the reverse triangle inequality, we have:

$$\begin{aligned}
&\mathscr{L}_k^{-1} \left| \|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\| - \|\boldsymbol{e}_k(\hat{\theta}_k)\| \right| \\
&\quad \leq \mathscr{L}_k^{-1} \|\boldsymbol{g}_k(\hat{\theta}_k)\| \\
&\quad \leq \|\hat{\theta}_k - \theta_k^*\| \\
&\quad \leq \mathscr{C}_k^{-1} \|\boldsymbol{g}_k(\hat{\theta}_k)\| \\
&\quad \leq \mathscr{C}_k^{-1} \left[\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\| + \|\boldsymbol{e}_k(\hat{\theta}_k)\| \right]. \qquad (3.46)
\end{aligned}$$

Taking conditional expectation over (3.46) and invoking A.1 yields (3.44). □

To obtain a conditional bound on the drift term, let us consider the following filtration instead of (2.3).

$$\mathcal{G}_0 = \mathcal{F}_0 = \sigma\{\hat{\theta}_0\}, \text{ and } \mathcal{G}_k = \sigma\{\hat{\theta}_0, \hat{\boldsymbol{g}}_i(\hat{\theta}_{i-1}), \hat{\boldsymbol{g}}_i(\hat{\theta}_i), i < k\} \text{ for } k \geq 1, \qquad (3.47)$$

which is finer (richer) than $\mathcal{F}_k$. We may have the following indicators for the tracking performance.

Theorem 3.3.4 (Estimation for drift using two-measurements)**.** *Under Assumptions A.1, A.2, and A.3, we have the following drift bound conditioned on $\mathcal{G}_{k+2}$:*

$$\mathbb{E}\left[\|\theta^*_{k+1}-\theta^*_k\|\middle|\,\mathcal{G}_{k+2}\right]\le \mathscr{C}_{k+1}^{-1}\|\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)\|+\mathscr{C}_k^{-1}\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|+\mathscr{M}_{k+1}\mathscr{C}_{k+1}^{-1}+\mathscr{M}_k\mathscr{C}_k^{-1}, \tag{3.48}$$

and

$$\begin{aligned}&\mathbb{E}\left[\|\theta^*_{k+1}-\theta^*_k\|\middle|\,\mathcal{G}_{k+2}\right]\\ &\quad\ge \max\Bigg\{\mathscr{L}_{k+1}^{-1}\left|\|\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)\|-\mathscr{M}_{k+1}\right|-\mathscr{C}_k^{-1}\left(\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|+\mathscr{M}_k\right),\\ &\qquad\qquad \mathscr{L}_k^{-1}\left|\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|-\mathscr{M}_k\right|-\mathscr{C}_{k+1}^{-1}\left(\|\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)\|+\mathscr{M}_{k+1}\right)\Bigg\}.\end{aligned} \tag{3.49}$$

Proof for Theorem 3.3.4.

$$\begin{aligned}\|\theta^*_{k+1}-\theta^*_k\| &\le \|\theta^*_{k+1}-\hat{\theta}_k\|+\|\hat{\theta}_k-\theta^*_k\|\\ &\le \mathscr{C}_{k+1}^{-1}\|\boldsymbol{g}_{k+1}(\hat{\theta}_k)\|+\mathscr{C}_k^{-1}\|\boldsymbol{g}_k(\hat{\theta}_k)\|\\ &\le \mathscr{C}_{k+1}^{-1}\left(\|\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)\|+\|\boldsymbol{e}_{k+1}(\hat{\theta}_k)\|\right)+\mathscr{C}_k^{-1}\left(\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|+\|\boldsymbol{e}_k(\hat{\theta}_k)\|\right),\end{aligned} \tag{3.50}$$

where the second inequality follows from (3.45). Taking the conditional expectation of (3.50) over $\mathcal{G}_{k+2}$ gives (3.48).

$$
\begin{aligned}
& \|\theta^*_{k+1} - \theta^*_k\| \\
\geq\ & \max\left\{\|\theta^*_{k+1} - \hat{\theta}_k\| - \|\theta^*_k - \hat{\theta}_k\|, \|\theta^*_k - \hat{\theta}_k\| - \|\theta^*_{k+1} - \hat{\theta}_k\|\right\} \\
\geq\ & \max\left\{\mathscr{L}_{k+1}^{-1}\|\boldsymbol{g}_{k+1}(\hat{\theta}_k)\| - \mathscr{C}_k^{-1}\|\boldsymbol{g}_k(\hat{\theta}_k)\|, \mathscr{L}_k^{-1}\|\boldsymbol{g}_k(\hat{\theta}_k)\| - \mathscr{C}_{k+1}^{-1}\|\boldsymbol{g}_{k+1}(\hat{\theta}_k)\|\right\} \\
\geq\ & \max\left\{\mathscr{L}_{k+1}^{-1}\left|\|\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)\| - \|\boldsymbol{e}_{k+1}(\hat{\theta}_k)\|\right| - \mathscr{C}_k^{-1}\left(\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\| + \|\boldsymbol{e}_k(\hat{\theta}_k)\|\right),\right. \\
& \qquad \left.\mathscr{L}_k^{-1}\left|\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\| - \|\boldsymbol{e}_k(\hat{\theta}_k)\|\right| - \mathscr{C}_{k+1}^{-1}\left(\|\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)\| + \|\boldsymbol{e}_{k+1}(\hat{\theta}_k)\|\right)\right\}, \qquad (3.51)
\end{aligned}
$$

where the second inequality follows from (3.45). Taking the conditional expectation of (3.51) over $\mathcal{G}_{k+2}$ gives (3.49). □

Corollary 3.3.1 (Conditional mean tracking performance)**.** *In addition to the conditions in Theorem 3.3.4, further assuming* $\mathbb{E}[\boldsymbol{e}_{k+1}(\hat{\theta}_k)\Big|\,\mathcal{G}_{k+2}] = \mathbf{0}$*. , we have:*

$$
\begin{aligned}
& \mathbb{E}\left[f_{k+1}(\hat{\theta}_k) - f_{k+1}(\theta^*_{k+1})\Big|\,\mathcal{G}_{k+2}\right] \\
& \quad \leq \frac{1}{2\mathscr{C}_{k+1}}\left(\|\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)\|^2 + \mathscr{M}_{k+1}^2\right) + \frac{a_k^2\mathscr{L}_{k+1}}{2}\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|^2 - a_k[\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\theta}_k),
\end{aligned} \qquad (3.52)
$$

and

$$\begin{aligned}
&\mathbb{E}\left[f_{k+1}(\hat{\theta}_k) - f_{k+1}(\theta^*_{k+1}) \middle| \mathcal{G}_{k+2} \right] \\
&\quad \geq \frac{1}{2\mathscr{L}_{k+1}} \|\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)\|^2 + \frac{a_k \mathscr{C}_{k+1}}{2} \|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|^2 - a_k [\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)]^T \hat{\boldsymbol{g}}_k(\hat{\theta}_k), \qquad (3.53)
\end{aligned}$$

where the expectation is conditioned on $\mathcal{G}_{k+2} = \sigma\{\hat{\theta}_0, \hat{\boldsymbol{g}}_i(\hat{\theta}_i), \hat{\boldsymbol{g}}_i(\hat{\theta}_{i-1}), i < k+2\}$.

Proof of Corollary 3.3.1. With the recursion in (2.1), we have the following from inequality (3.12):

$$\begin{aligned}
&f_{k+1}(\hat{\theta}_{k+1}) \\
&\quad = f_{k+1}(\hat{\theta}_k - a_k \hat{\boldsymbol{g}}_k(\hat{\theta}_k)) \\
&\quad \leq f_{k+1}(\hat{\theta}_k) - a_k [\boldsymbol{g}_{k+1}(\hat{\theta}_k)]^T \hat{\boldsymbol{g}}_k(\hat{\theta}_k) + \frac{\mathscr{L}_{k+1}}{2} a_k^2 \|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|^2 \\
&\quad = f_{k+1}(\hat{\theta}_k) - a_k [\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)]^T \hat{\boldsymbol{g}}_k(\hat{\theta}_k) + a_k [\boldsymbol{e}_{k+1}(\hat{\theta}_k)]^T \hat{\boldsymbol{g}}_k(\hat{\theta}_k) + \frac{a_k^2 \mathscr{L}_{k+1}}{2} \|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|^2.
\end{aligned} \qquad (3.54)$$

Therefore,

$$\begin{aligned}
&\mathbb{E}\left[f_{k+1}(\hat{\boldsymbol{\theta}}_{k+1}) - f_{k+1}(\boldsymbol{\theta}^*_{k+1})\middle|\,\mathcal{G}_{k+2}\right] \\
&\quad\le \mathbb{E}\left[f_{k+1}(\hat{\boldsymbol{\theta}}_{k}) - f_{k+1}(\boldsymbol{\theta}^*_{k+1})\middle|\,\mathcal{G}_{k+2}\right] - a_k[\hat{\boldsymbol{g}}_{k+1}(\hat{\boldsymbol{\theta}}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) + \frac{a_k^2\mathscr{L}_{k+1}}{2}\|\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)\|^2 \\
&\quad\le \frac{1}{2\mathscr{C}_{k+1}}\mathbb{E}\left[\|\boldsymbol{g}_{k+1}(\hat{\boldsymbol{\theta}}_k)\|^2\middle|\,\mathcal{G}_{k+2}\right] + \frac{a_k^2\mathscr{L}_{k+1}}{2}\|\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)\|^2 - a_k[\hat{\boldsymbol{g}}_{k+1}(\hat{\boldsymbol{\theta}}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) \\
&\quad\le \frac{1}{2\mathscr{C}_{k+1}}\left(\|\hat{\boldsymbol{g}}_{k+1}(\hat{\boldsymbol{\theta}}_k)\|^2 + \mathscr{M}_{k+1}^2\right) + \frac{a_k^2\mathscr{L}_{k+1}}{2}\|\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)\|^2 - a_k[\hat{\boldsymbol{g}}_{k+1}(\hat{\boldsymbol{\theta}}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k),
\end{aligned} \tag{3.55}$$

where the second last inequality follows from (3.8).

Similarly, from the recursion in (2.1), we have the following from inequality (3.11):

$$\begin{aligned}
&f_{k+1}(\hat{\boldsymbol{\theta}}_{k+1}) \\
&\quad\ge f_{k+1}(\hat{\boldsymbol{\theta}}_k) - a_k[\boldsymbol{g}_{k+1}(\hat{\boldsymbol{\theta}}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) + \frac{\mathscr{C}_{k+1}}{2}a_k^2\|\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)\|^2 \\
&\quad= f_{k+1}(\hat{\boldsymbol{\theta}}_k) - a_k[\hat{\boldsymbol{g}}_{k+1}(\hat{\boldsymbol{\theta}}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) + a_k[\boldsymbol{e}_{k+1}(\hat{\boldsymbol{\theta}}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) + \frac{a_k^2\mathscr{C}_{k+1}}{2}\|\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)\|^2.
\end{aligned} \tag{3.56}$$

Therefore,

$$\begin{aligned}
&\mathbb{E}\left[f_{k+1}(\hat{\theta}_{k+1}) - f_{k+1}(\theta^*_{k+1})\middle|\,\mathcal{G}_{k+2}\right] \\
&\quad\geq \mathbb{E}\left[f_{k+1}(\hat{\theta}_k) - f_{k+1}(\theta^*_{k+1})\middle|\,\mathcal{G}_{k+2}\right] - a_k[\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\theta}_k) + \frac{a_k^2\mathscr{C}_{k+1}}{2}\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|^2 \\
&\quad\geq \frac{1}{2\mathscr{L}_{k+1}}\mathbb{E}\left[\|\boldsymbol{g}_{k+1}(\hat{\theta}_k)\|^2\middle|\,\mathcal{G}_{k+2}\right] + \frac{a_k\mathscr{C}_{k+1}}{2}\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|^2 - a_k[\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\theta}_k) \\
&\quad\geq \frac{1}{2\mathscr{L}_{k+1}}\|\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)\|^2 + \frac{a_k\mathscr{C}_{k+1}}{2}\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|^2 - a_k[\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_k)]^T\hat{\boldsymbol{g}}_k(\hat{\theta}_k). \qquad (3.57)
\end{aligned}$$

□

One Quick Example

Again, we use the similar setup as in Subsubsection 3.3.2, except that the evolution of $\{\theta^*_k\}$ in (3.43) now changes to:

$$\theta^*_{k+1} = \begin{cases} \theta^*_k + (1,\ 1)^T + \boldsymbol{w}_k, & \text{for } 1 \leq k \leq 499, \\ \theta^*_k + 200(\cos(\varphi),\ \sin(\varphi))^T, & \text{for } k = 500, \\ \theta^*_k + (-1,\ 1)^T + \boldsymbol{w}_k, & \text{for } 501 \leq k \leq 999, \end{cases} \qquad (3.58)$$

with $\theta^*_0 = \mathbf{0}$. Again, $\boldsymbol{w}_k \overset{\text{i.i.d.}}{\sim} \text{Normal}(\mathbf{0}, \sigma_2^2\boldsymbol{I}_p)$, and $\varphi \sim \text{Uniform}\,[0, 2\pi]$. All the other parameters and the gain sequence selection remain the same as Subsubsection 3.3.2.

Still following the general procedure in Algorithm 1, this time we do not know $\mathscr{B}_k$ a priori and cannot proceed with the computation of the error bounds established in

Subsection 3.3.2. Figure 3.4 displays how the iterates generated by (2.1) keep up with the moving target using the same gain sequence as the one used in Subsection 3.3.2. Figure 3.5 shows how collecting two measurements at a time, i.e., collecting (3.47), to bounding the drift term $\|\theta_{k+1}^* - \theta_k^*\|$ per (3.48) in nonstationary optimization.

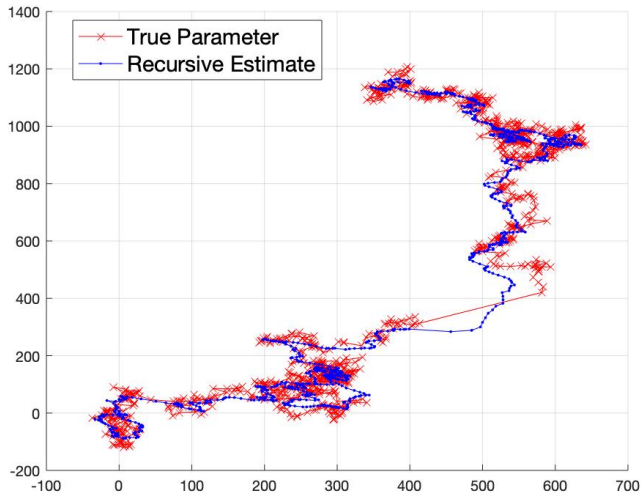


Figure 3.4: The Trajectories Of The Underlying Parameter θ_k^* And The Recursive Estimates $\hat{\theta}_k$ In *One* Run

Call For A Data-Driven Gain-Tuning Strategy

Under the general time-varying assumption A.4, note that (3.48) can provide a rudimentary assessment to the variation in $\|\theta_{k+1}^* - \theta_k^*\|$, and that (3.44) can help bounding the conditional MAD. However, our gain selection strategy in Lemma 3.3.6 (equivalently Algorithm 1) is developed on the basis that $\mathscr{M}_k$ in A.1 and $\mathscr{B}_k$ in A.4 are relatively small and on the purpose of ensuring *average* tracking performance as opposed to a single sample-path. Therefore, we may need to develop a new gain selection strategy to be

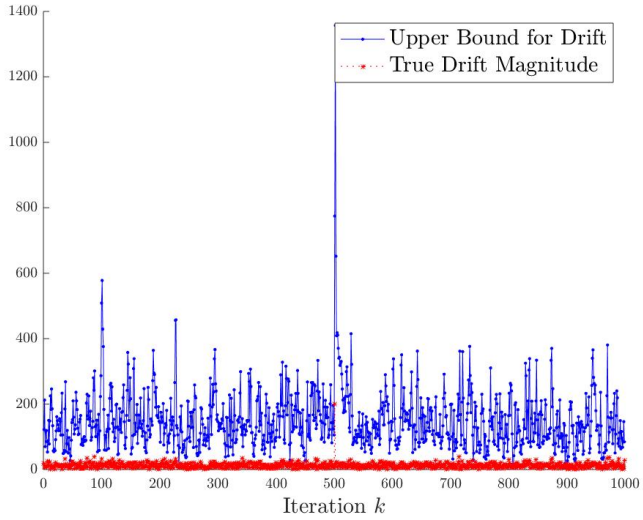


Figure 3.5: Comparison Between The Upper Bound (3.48) Obtained From Two Measurements And The Actual Drift Term $\|\theta^*_{k+1} - \theta^*_k\|$

relatively robust to abrupt changes as indicated by $\mathbb{E}_{k+2}\|\theta^*_{k+1} - \theta^*_k\|$, if any such change arises within a particular sample-path. See further details in Chapter 5.

Observe from Theorem 3.3.3 that a larger value of $\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|$ is strong evidence that $\hat{\theta}_k$ is further away from θ^*_k. However, within one run of generating $\hat{\theta}_0, \cdots, \hat{\theta}_k$, we *cannot* differentiate whether or not the large value of $\|\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\|$ is due to excessive noise or due to the abrupt jump from θ^*_k to θ^*_{k+1}. Furthermore, as mentioned in Subsection 3.1.3, we do not want to consider "multiple *sequential* measurements at a time," especially an excessive number of measurements (same order of the squared of the inverse desired accuracy) as in [117].

Therefore, we will turn to a more restrictive scenario in Chapter 5. It is desirable to obtain a testing rule under which SA iterates can promptly detect the change in $\{\theta^*_k\}$ and provide guidance in gain selection. It is certainly advantageous to use adaptive rules that

enable the stepsize to vary with information gathered during the progress of the estimation procedure.

3.4 Special Cases

3.4.1 Regression with Time-Varying Underlying Parameter

Least-Mean-Squares

In the linear regression model, we assume the following measurement equation that is linear in θ_k^*:

$$z_k = \boldsymbol{h}_k^T \theta_k^* + v_k, \ k \in \mathbb{N}, \tag{3.59}$$

where z_k is the kth scalar measurement of the output, $\boldsymbol{h}_k$ is a $p \times 1$ stochastic design vector of the input or regression vector, θ_k^* is the underlying target parameter, which evolves smoothly along the passage of time, and v_k is a mean-zero disturbance sequence. For multiple-input-single-output (MISO) system (3.59), the goal is to use known input values of $\boldsymbol{h}_k$ (e.g., from a training sequence) and observed output values z_k to estimate and track the underlying MISO system parameter θ_k^*. The time-varying function we are trying to minimize is

$$f_k(\theta) = \frac{1}{2}\mathbb{E}\left[(z_{k+1} - \boldsymbol{h}_{k+1}^T \theta)^2\right], \tag{3.60}$$

where the expectation is taken w.r.t. the noise v_{k+1} in (3.59) and the randomness in $\boldsymbol{h}_{k+1}$ if the $\boldsymbol{h}_{k+1}$ is random. Here $\boldsymbol{h}_k$ is the controllable input (may be random), while z_k is the output that contains partial information on $\boldsymbol{\theta}_k^*$.

Suppose that the measurement noise v_k is independent of both $\boldsymbol{\theta}$ and $\boldsymbol{h}_k$. Then the derivative of the time-varying loss function (3.60) w.r.t. parameter $\boldsymbol{\theta}$ is

$$\begin{aligned}
\boldsymbol{g}_k(\boldsymbol{\theta}) &= \frac{1}{2}\frac{\partial \mathbb{E}\left[(z_{k+1}-\boldsymbol{h}_{k+1}^T\boldsymbol{\theta})^2\right]}{\partial \boldsymbol{\theta}} \\
&= \frac{1}{2}\frac{\partial \mathbb{E}\left[(\boldsymbol{h}_{k+1}^T\boldsymbol{\theta}_{k+1}^*+v_{k+1}-\boldsymbol{h}_{k+1}^T\boldsymbol{\theta})^2\right]}{\partial \boldsymbol{\theta}} \\
&= \begin{cases} \boldsymbol{h}_{k+1}\boldsymbol{h}_{k+1}^T(\boldsymbol{\theta}-\boldsymbol{\theta}_{k+1}^*), \text{ when } \boldsymbol{h}_{k+1} \text{ is deterministic,} \\ \mathbb{E}(\boldsymbol{h}_{k+1}\boldsymbol{h}_{k+1}^T)(\boldsymbol{\theta}-\boldsymbol{\theta}_{k+1}^*), \text{ when } \boldsymbol{h}_{k+1} \text{ is stochastic yet independent of } v_{k+1}. \end{cases}
\end{aligned} \tag{3.61}$$

Note that the expectation in the last line is w.r.t. the input-noise pair $(\boldsymbol{h}_{k+1}, v_{k+1})$. The randomness in $\boldsymbol{\theta}_{k+1}^*$, if there is any, is not involved (note that the computation of (3.60) and (3.61) is infeasible in reality due to the unavailability to carry out the expectation in (3.60) and the unknown target $\boldsymbol{\theta}_{k+1}^*$). The most accessible information is the instantaneous gradient:

$$\begin{aligned}
\hat{\boldsymbol{g}}_k(\boldsymbol{\theta}) &\equiv \frac{1}{2}\frac{\partial[\left(z_{k+1}-\boldsymbol{h}_{k+1}^T\boldsymbol{\theta}\right)^2]}{\partial \boldsymbol{\theta}} \\
&= \boldsymbol{h}_{k+1}(\boldsymbol{h}_{k+1}^T\boldsymbol{\theta}-z_{k+1}).
\end{aligned} \tag{3.62}$$

Estimate (3.62) is a stochastic gradient due to the derivative of the argument inside the expectation operator in (3.60). Besides, (3.62) is an unbiased estimator of (3.61).

When applying in linear regression models, $\boldsymbol{h}_k$ represents the gradient of the predicted model output w.r.t. the parameter θ in the model (3.59), and then the recursion (2.1) reduces to the LMS algorithm. Explicitly, the stochastic gradient at step k is calculated as:

$$\begin{aligned} \hat{\theta}_{k+1} &= \hat{\theta}_k - a_{k+1}\boldsymbol{h}_{k+1}(\boldsymbol{h}_{k+1}^T\hat{\theta}_k - z_{k+1}) \\ &= (\boldsymbol{I}_p - a\boldsymbol{h}_{k+1}\boldsymbol{h}_{k+1}^T)\hat{\theta}_k + a_{k+1}\boldsymbol{h}_{k+1}z_{k+1}, \ \ k \in \mathbb{N}, \end{aligned} \tag{3.63}$$

where $\hat{\theta}_0$ is chosen arbitrarily or with a priori information, and is assumed to have a finite second moment.

Comparing (3.61) and (3.62), the error term in (3.5) becomes:

$$\boldsymbol{e}_k(\theta) = \begin{cases} -\boldsymbol{h}_{k+1}v_{k+1}, \quad \text{when } \boldsymbol{h}_{k+1} \text{ is deterministic,} \\ [\boldsymbol{h}_{k+1}\boldsymbol{h}_{k+1}^T - \mathbb{E}(\boldsymbol{h}_{k+1}\boldsymbol{h}_{k+1}^T)](\theta - \theta_{k+1}^*) - \boldsymbol{h}_{k+1}v_{k+1}, \\ \qquad \text{when } \boldsymbol{h}_{k+1} \text{ is stochastic yet independent of } v_{k+1}. \end{cases} \tag{3.64}$$

Here $\boldsymbol{e}_k(\theta)$ is mean-zero as long as the measurement noise v_k in (3.59) is mean-zero (as assumed above).

Remark 5. The change of θ_k^* is called state evolution. Naturally, all the randomnesses in the dynamic system, which consists of (3.59) and the state evolution, arise from the $\{\theta_k^*\}$ in the state evolution and the input-noise pair $\{\boldsymbol{h}_k, v_k\}$ in the measurement equation. Note that under A.4, the sequence $\{\theta_k^*\}_{k\geq 0}$ is allowed to be either stochastic or fully deterministic.

Note that Assumptions A.1–A.4 listed previously for (2.1) can be specialized to the LMS algorithm (3.63) as in [122]. The required assumptions are (1) the design vector sequence $\{\boldsymbol{h}_k\}$ is random[5] and has a bounded L_2 norm uniformly across k, (2) v_k in (3.59) is mean-zero and has a bounded variance of $\sigma_{v_k}^2$, and (3) the pair $\{\boldsymbol{h}_k, v_k\}$ is independent of θ_k^*. Immediately, $\mathscr{M}_k$ in A.1 becomes $\sqrt{\sigma_{v_{k+1}}^2(\mathbb{E}[\|\boldsymbol{h}_{k+1}\|^2])}$, $\mathscr{C}_k$ in A.2 becomes $\lambda_{\min}(\mathbb{E}(\boldsymbol{h}_{k+1}\boldsymbol{h}_{k+1}^T))$, and $\mathscr{L}_k$ in A.3 becomes $\lambda_{\max}(\mathbb{E}(\boldsymbol{h}_{k+1}\boldsymbol{h}_{k+1}^T))$.

Prior work on error bounds for the linear case include [37, 44, 74]. However, the bounds therein are usually not computable, as they require higher-order (higher than second-order) moments information of the design vector $\boldsymbol{h}_k$. Admittedly, the error bound (3.31) and (3.36) also requires information regarding $\mathbb{E}(\boldsymbol{h}_k\boldsymbol{h}_k^T)$, but the estimation of $\mathbb{E}(\boldsymbol{h}_k\boldsymbol{h}_k^T)$ on the fly requires 2nd-oder information and that $\boldsymbol{h}_k$'s are i.i.d. As explained in Subsection 3.2.1, we do not dwell on the estimation issues given that our problem setup only allows a *few* observations. Also, there are numerous works on the random-walk evolvement assumption (based on a linear model, mainly for LMS): [71] and [99, Chap. 5], but they are not as informative and general as our results (3.32) and (3.37) that reveal the dependency explicitly on the gain selection, the noise level, the drift level, and the second-order information.

[5]For the case where $\boldsymbol{h}_k$ is deterministic, see [43]

General Empirical Risk Minimization

In general empirical risk minimization, given data pairs $(\boldsymbol{h}_k, \boldsymbol{z}_k)$, we wish to learn a hypothesized relationship $\boldsymbol{z}_k \approx \boldsymbol{\varphi}(\boldsymbol{h}_k)$ for $\boldsymbol{\varphi}$ chosen from a family of functions $\{\boldsymbol{\varphi}_\theta\}$ parametrized[6] by $\theta \in \mathbb{R}^p$. That is, (3.59) becomes

$$\boldsymbol{z}_k = \boldsymbol{\varphi}_{\theta_k^*}(\boldsymbol{h}_k) + \boldsymbol{v}_k, \quad k \in \mathbb{N} \tag{3.65}$$

where θ_k^* is the underlying target parameter which evolves smoothly along the passage of time, and $\boldsymbol{v}_k$ is a disturbance sequence with a mean of $\mathbf{0}$. Note that the function form of $\boldsymbol{\varphi}_\theta(\cdot)$ allows for both the linear representation as in (3.59) and nonlinear form, and $\boldsymbol{h}_k$ is not necessarily in $\mathbb{R}^p$ due to the potentially nonlinear mapping $\boldsymbol{\varphi}_\theta(\cdot)$. For the multiple-input-multiple-output (MIMO) system (3.65), we aim to use the known input values $\boldsymbol{h}_k$ and observed output values $\boldsymbol{z}_k$ to estimate and track the underlying MIMO system parameter θ_k^*. The time-varying function we are trying to minimize is

$$f_k(\theta) = \frac{1}{2}\mathbb{E}\left[\|\boldsymbol{z}_{k+1} - \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})\|^2\right], \tag{3.66}$$

where the expectation in (3.66) is taken w.r.t. the noise $\boldsymbol{v}_k$ in (3.65) and the randomness in $\boldsymbol{h}_{k+1}$ if $\boldsymbol{h}_{k+1}$ is random. Here $\boldsymbol{h}_k$ is the controllable input which may be random, while $\boldsymbol{z}_k$ is the output that contains partial information on θ_k^*.

[6]By "parametrized" we mean that the mapping from θ to $\boldsymbol{\varphi}_\theta(\cdot)$ is one-to-one. Specifically, $\theta_1 \neq \theta_2$ implies $\boldsymbol{\varphi}_{\theta_1}(\cdot) \neq \boldsymbol{\varphi}_{\theta_2}(\cdot)$. Alternatively, $\boldsymbol{\varphi}_{\theta_1}(\cdot) = \boldsymbol{\varphi}_{\theta_2}(\cdot)$ implies $\theta_1 = \theta_2$.

Suppose that the measurement noise $\boldsymbol{v}_k$ in (3.65) is independent of both θ and $\boldsymbol{h}_k$, then the derivative of the time-varying loss function (3.66) w.r.t. θ is

$$\begin{aligned}
\boldsymbol{g}_k(\theta) &= -\mathbb{E}\left[\left(\frac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T (\boldsymbol{z}_{k+1} - \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1}))\right] \\
&= \mathbb{E}\left[\left(\frac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T \left(\boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1}) - \boldsymbol{\varphi}_{\theta^*_{k+1}}(\boldsymbol{h}_{k+1}) - \boldsymbol{v}_{k+1}\right)\right] \\
&= \begin{cases}
\left(\dfrac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T (\boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1}) - \boldsymbol{\varphi}_{\theta^*_{k+1}}(\boldsymbol{h}_{k+1})), & \text{when } \boldsymbol{h}_{k+1} \text{ is deterministic,} \\
\mathbb{E}\left[\left(\dfrac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T \left(\boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1}) - \boldsymbol{\varphi}_{\theta^*_{k+1}}(\boldsymbol{h}_{k+1})\right)\right], & \\
\qquad\qquad \text{when } \boldsymbol{h}_{k+1} \text{ is stochastic yet independent of } \boldsymbol{v}_{k+1},
\end{cases}
\end{aligned} \tag{3.67}$$

where we have assumed that the differentiation interchanges with the integral (expectation), and the expectation is w.r.t. the data pair $(\boldsymbol{h}_{k+1}, \boldsymbol{z}_{k+1})$. The randomness in θ^*_{k+1}, if there is any, is not involved. Oftentimes, the joint distribution of $(\boldsymbol{h}_k, \boldsymbol{z}_k)$ is unknown. The accessible information is the instantaneous gradient:

$$\hat{\boldsymbol{g}}_k(\theta) = \left(\frac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T (\boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1}) - \boldsymbol{z}_{k+1}). \tag{3.68}$$

Hence, SA recursion at step k is calculated as:

$$\begin{aligned}
\hat{\theta}_{k+1} &= \hat{\theta}_k - a_k \hat{\boldsymbol{g}}_k(\hat{\theta}_k) \\
&= \hat{\theta}_k - a_k \left(\frac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\bigg|_{\theta=\hat{\theta}_k}\right)^T (\boldsymbol{\varphi}_{\hat{\theta}_k}(\boldsymbol{h}_{k+1}) - \boldsymbol{z}_{k+1}), \quad k \in \mathbb{N}
\end{aligned} \tag{3.69}$$

with an initialization $\hat{\theta}_0$ being deterministic or stochastic but with the finite second moment. We immediately see that the LMS algorithm (3.63) is a special case of the general principle of empirical risk minimization (3.69). Comparing (3.67) and (3.68), the error term in (3.5) becomes

$$\boldsymbol{e}_k(\theta) = \begin{cases} -\left(\dfrac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T \boldsymbol{v}_{k+1}, \text{ when } \boldsymbol{h}_{k+1} \text{ is deterministic} \\ \left[\left(\dfrac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T \left(\boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1}) - \boldsymbol{\varphi}_{\theta^*_{k+1}}(\boldsymbol{h}_{k+1})\right)\right] \\ \quad -\mathbb{E}\left[\left(\dfrac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T \left(\boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1}) - \boldsymbol{\varphi}_{\theta^*_{k+1}}(\boldsymbol{h}_{k+1})\right)\right] - \left(\dfrac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T \boldsymbol{v}_{k+1}, \\ \qquad\qquad \text{when } \boldsymbol{h}_{k+1} \text{ is stochastic yet independent of } \boldsymbol{v}_{k+1}, \end{cases}$$

Here $\boldsymbol{e}_k(\theta)$ has a mean of $\mathbf{0}$ as long as the measurement noise $\boldsymbol{v}_k$ in (3.65) has a mean of $\mathbf{0}$.

Now the Assumptions A.1—A.4 listed for the general SA algorithm (2.1) can be specialized for (3.68). The required assumptions are (1) the family of functions $\{\boldsymbol{\varphi}_\theta\}$ is parametrized by θ, and every the second-order partial derivatives of $\boldsymbol{\varphi}_\theta(\cdot)$ w.r.t. θ, which is a 3-dimensional matrix (a.k.a. tensor), are continuous in θ, (2) $\boldsymbol{v}_k$ in (3.65) has a mean of $\mathbf{0}$ and a covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{v}_k}$ with bounded entries, (3) the pair $\{\boldsymbol{h}_k, \boldsymbol{v}_k\}$ is independent of θ^*_k. Immediately, $\mathscr{M}_k$ in A.1 becomes

$$\sup_{\theta \in \mathbb{R}^p} \sqrt{\mathrm{tr}\left(\left(\frac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)\left(\frac{\partial \boldsymbol{\varphi}_\theta(\boldsymbol{h}_{k+1})}{\partial \theta}\right)^T \boldsymbol{\Sigma}_{\boldsymbol{v}_{k+1}}\right)},$$

$\mathscr{C}_k$ in A.2 and $\mathscr{L}_k$ in A.3 become $\inf_{\theta} \lambda_{\min}[\partial \boldsymbol{g}_k(\theta)/\partial\theta]$ and $\sup_{\theta} \lambda_{\max}[\partial \boldsymbol{g}_k(\theta)/\partial\theta]$, where $\boldsymbol{g}_k(\cdot)$ is defined in (3.67). We omit the detailed expression here as it involves the notion of tensor and the definition of multiplying a tensor by a matrix, which is not the focus here.

3.4.2 General Adaptive Algorithms

Subsection 3.4.1 discusses the scenario where the $\{\theta_k^*\}$ evolution is unknown, and the general form of SA algorithm (2.1) is used to track the time variation. Nonetheless, if the evolution law is partially revealed, it should be taken into consideration in the time-varying parameter estimation along the lines of [106, Eq. (3.19) on p. 84]. For example, the prediction step in KF, which is similar to (3.82) to appear, makes direct use of the linear state-space model.

Static Kalman Filtering

In general, the classical KF algorithm cannot be rearranged as a special case of (2.1). Here is an exception: consider the case when there are no dynamics, that is,

$$\begin{cases} \text{Static Model: } \theta_k^* = \theta^*, \\ \text{Measurement: } \boldsymbol{z}_k = \boldsymbol{H}_k \theta_k^* + \boldsymbol{v}_k, \end{cases} \tag{3.70}$$

and where the observation $\boldsymbol{z}_k \in \mathbb{R}^{p'}$ (usually $1 \leq p' \ll p$), the matrix $\boldsymbol{H}_k \in \mathbb{R}^{p' \times p}$ is known, the independent sequence $\{\boldsymbol{v}_k\}$ satisfies $\mathbb{E}(\boldsymbol{v}_k) = \boldsymbol{0}$ and $\mathbb{C}(\boldsymbol{v}_k) = \boldsymbol{R}_k$ (which is symmetric). Moreover, θ_0^* is random with a known mean and a known variance $\boldsymbol{P}_0$.

Remark 6. The general framework of Kalman filtering that allows time-varying θ_k^* (a nonzero $\mathscr{B}_k$) pertains to second-order derivative w.r.t. θ and second-order noise statistics for both the modeling noise and the measurement noise, and cannot be put into the first-order SA algorithm framework (2.1). Hence it is not discussed here.

Then the prediction of the state estimation and the covariance estimate from the KF are

$$\begin{cases} \hat{\theta}_{k|k-1} = \hat{\theta}_{k-1}, \quad \text{with } \hat{\theta}_0 = \mathbb{E}(\theta_0), \\ \boldsymbol{P}_{k|k-1} = \boldsymbol{P}_{k-1}, \quad \text{with } \boldsymbol{P}_0 = \mathbb{E}[(\hat{\theta}_0 - \theta_0)(\hat{\theta}_0 - \theta_0)^T], \end{cases} \tag{3.71}$$

and the updating step is

$$\begin{cases} \boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T(\boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T + \boldsymbol{R}_k)^{-1}, & (3.72) \\ \hat{\theta}_k = \hat{\theta}_{k|k-1} + \boldsymbol{K}_k(\boldsymbol{z}_k - \boldsymbol{H}_k\hat{\theta}_{k|k-1}), & (3.73) \\ \boldsymbol{P}_k = (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_{k|k-1}. & (3.74) \end{cases}$$

Above updating formulas imply:

$$\boldsymbol{P}_{k+1}^{-1} = \boldsymbol{P}_k^{-1} + \boldsymbol{H}_k^T\boldsymbol{R}_k^{-1}\boldsymbol{H}_k \quad \text{and} \quad \boldsymbol{K}_k = \boldsymbol{P}_k\boldsymbol{H}_k^T\boldsymbol{R}_k^{-1}. \tag{3.75}$$

For MIMO system (3.70), the time-varying function we are trying to minimize at each sampling instance τ_k is

$$f_k(\theta) = \mathbb{E}\left[\frac{1}{2}(\boldsymbol{z}_{k+1} - \boldsymbol{H}_{k+1}\theta)^T\boldsymbol{R}_{k+1}^{-1}(\boldsymbol{z}_{k+1} - \boldsymbol{H}_{k+1}\theta)\right], \tag{3.76}$$

where the expectation is taken w.r.t. the noise $\boldsymbol{v}_k$ in (3.70). Suppose that the measurement noise $\boldsymbol{v}_k$ in (3.70) is independent of the valuation point θ, then the *instantaneous* gradient of the time-varying loss function (3.76) can be obtained by taking the derivative of the quantity inside of the expectation operator in (3.76) w.r.t. θ:

$$\begin{aligned}\hat{\boldsymbol{g}}_k(\theta) &= \frac{1}{2}\frac{\partial[(\boldsymbol{z}_{k+1}-\boldsymbol{H}_{k+1}\theta)^T\boldsymbol{R}_{k+1}^{-1}(\boldsymbol{z}_{k+1}-\boldsymbol{H}_{k+1}\theta)]}{\partial\theta}\\ &= \boldsymbol{H}_{k+1}^T\boldsymbol{R}_{k+1}^{-1}(\boldsymbol{H}_{k+1}\theta-\boldsymbol{z}_{k+1})\\ &= \boldsymbol{P}_{k+1}^{-1}\boldsymbol{K}_{k+1}(\boldsymbol{H}_{k+1}\theta-\boldsymbol{z}_{k+1}),\end{aligned} \tag{3.77}$$

whose error term defined in (3.5) is

$$\boldsymbol{e}_k(\theta)=\hat{\boldsymbol{g}}_k(\theta)-\boldsymbol{g}_k(\theta)=\boldsymbol{H}_{k+1}^T\boldsymbol{R}_{k+1}^{-1}\boldsymbol{H}_{k+1}(\theta-\theta_{k+1}^*)-\boldsymbol{H}_{k+1}^T\boldsymbol{R}_{k+1}^{-1}\boldsymbol{v}_{k+1}. \tag{3.78}$$

Remark 7. The loss function construction (3.76), the stochastic gradient form (3.77), and the error form (3.78) for MIMO model (3.70) are natural extensions of (3.60), (3.62), and (3.64) for MISO model (3.59).

Then (3.73) becomes

$$\begin{aligned}\hat{\theta}_{k+1} &= \hat{\theta}_k+\boldsymbol{K}_{k+1}(\boldsymbol{z}_{k+1}-\boldsymbol{H}_{k+1}\hat{\theta}_k)\\ &= \theta_k-\hat{\boldsymbol{P}}_{k+1}\hat{\boldsymbol{g}}_k(\hat{\theta}_k),\end{aligned} \tag{3.79}$$

which aligns with the SGD algorithm (2.1) where $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ is replaced by (2.5), except that the scalar gain a_k is replaced by the matrix gain $\hat{\boldsymbol{P}}_{k+1}$.

Similar to Subsection 3.4.1, the assumptions A.1–A.4 can be specialized for the static Kalman filter algorithm (3.73). That is, as long as the observation matrix $\boldsymbol{H}_k \in \mathbb{R}^{p' \times p}$ has full (row) rank and has a bounded ℓ_2 norm uniformly across k, $\boldsymbol{v}_k$ in (3.70) is mean-zero and has a nonsingular[7] covariance matrix $\boldsymbol{R}_k$ uniformly for all k, and $\boldsymbol{v}_k$ is independent of θ_k^*. Specifically, $\mathscr{M}_k$ in A.1 becomes $\sqrt{\mathrm{tr}(\boldsymbol{H}_{k+1}^T \boldsymbol{R}_{k+1}^{-T} \boldsymbol{H}_{k+1})}$, $\mathscr{C}_k$ in A.2 becomes $\lambda_1(\boldsymbol{H}_{k+1}^T \boldsymbol{R}_{k+1}^{-1} \boldsymbol{H}_{k+1})$, $\mathscr{L}_k$ in A.3 becomes $\lambda_p(\boldsymbol{H}_{k+1}^T \boldsymbol{R}_{k+1}^{-1} \boldsymbol{H}_{k+1})$, and $\mathscr{B}_k$ in A.4 reduces to 0 in the static model (3.70).

General Dynamic Model With *Known* Evolution

As mentioned before, the classical KF and extended KF (EKF) algorithm cannot be rearranged as a special case of (2.1). Here we mention a simple tracking algorithm (3.83) that does not deal with the matrix multiplication and matrix inversion arising in computing the Kalman gain in KF/EKF, and provides a tracking error bound in Proposition 1.

In many applications, we hope to estimate a time-varying quantity that evolves with time according to a nonlinear state equation:

$$\theta_{k+1}^* = \boldsymbol{f}_k(\theta_k^*) + \boldsymbol{w}_k, \quad k = 0, 1, 2, \cdots, \tag{3.80}$$

[7] If the observations are nearly perfect, then $\boldsymbol{R}_k$ is close to $\boldsymbol{0}$. We do not dive into schemes in handling the consequent computational instability here.

where the evolution function form of $\boldsymbol{f}_k(\cdot) : \mathbb{R}^p \mapsto \mathbb{R}^p$ is *known*, and $\boldsymbol{w}_k$ is a mean-zero stochastic process. The incomplete information about the θ_k^* is available through observations $\boldsymbol{z}_k$ in the following form:

$$\boldsymbol{z}_k = \boldsymbol{h}_k(\theta_k^*) + \boldsymbol{v}_k, \quad k = 1, 2, \cdots, \tag{3.81}$$

where the measurement function form of $\boldsymbol{h}_k(\cdot) : \mathbb{R}^p \mapsto \mathbb{R}^p$ is *known*, and $\boldsymbol{v}_k$ is a mean-zero stochastic process. Equation (3.81) is naturally a nonlinear extension of (3.70). Following [106, Eq. (3.19) on p. 84], we proceed the time-varying parameter estimation via

$$\begin{cases} \text{Prediction step:} & \hat{\theta}_{k+1|k} = \boldsymbol{f}_k(\hat{\theta}_k), \\ \text{Updating step:} & \hat{\theta}_{k+1} = \hat{\theta}_{k+1|k} + a_{k+1}(\boldsymbol{z}_{k+1} - \boldsymbol{h}_{k+1}(\hat{\theta}_{k+1|k})), \end{cases} \tag{3.82}$$

where a_{k+1} is a scalar gain satisfying certain conditions. Combined, the recursion for $\hat{\theta}_k$ is:

$$\hat{\theta}_{k+1} = \boldsymbol{f}_k(\hat{\theta}_k) + a_{k+1}[\boldsymbol{z}_{k+1} - \boldsymbol{h}_{k+1}(\boldsymbol{f}_k(\hat{\theta}_k))], \quad k = 0, 1, 2, \cdots. \tag{3.83}$$

Proposition 1. *Consider the state equation (3.80) and the measurement equation (3.81). Assume that the following conditions hold.*

(a) The noise process $\boldsymbol{w}_k$ has a mean of $\mathbf{0}$ and a covariance matrix of $\boldsymbol{Q}_k$. The sequence $\{\boldsymbol{w}_k\}$ is an independent sequence.

(b) The noise process $\boldsymbol{v}_k$ has a mean of $\mathbf{0}$ and a covariance matrix of $\boldsymbol{R}_k$. The sequence $\{\boldsymbol{v}_k\}$ is an independent sequence.

(c) All the noises in $\{\boldsymbol{w}_k\}$ are independent of all the noises in $\{\boldsymbol{v}_k\}$.

(d) For every k, the squared-matrix-valued functions of θ, $d\boldsymbol{f}_k(\theta)/d\theta^T \equiv \dot{\boldsymbol{f}}_k(\theta)$ and $d\boldsymbol{h}_k(\theta)/d\theta^T \equiv \dot{\boldsymbol{h}}_k(\theta)$, are continuous w.r.t. θ.

(e) $\dot{\boldsymbol{f}}_k(\theta)$ is either positive-definite or negative definite for all θ. This also holds for $\dot{\boldsymbol{h}}_k(\theta)$ and $\boldsymbol{h}_{k+1}(\boldsymbol{f}_k(\theta))$.

When discussing positive/negative definiteness, we may write $\dot{\boldsymbol{f}}_k$ ($\dot{\boldsymbol{h}}_k$) instead of $\dot{\boldsymbol{f}}_k(\theta)$ ($\dot{\boldsymbol{h}}_k(\theta)$). Following the rational explained in Subsection 3.2.6, denote $\mathscr{L}_k^{\boldsymbol{f}} \equiv \lambda_p(|\dot{\boldsymbol{f}}_k|)$, $\mathscr{C}_k^{\boldsymbol{f}} \equiv \lambda_1(|\dot{\boldsymbol{f}}_k|)$, $\mathscr{R}_k^{\boldsymbol{f}} \equiv \mathscr{L}_k^{\boldsymbol{f}}/\mathscr{C}_k^{\boldsymbol{f}}$ Similarly define $\mathscr{L}_k^{\boldsymbol{h}}$, $\mathscr{C}_k^{\boldsymbol{h}}$, and $\mathscr{R}_k^{\boldsymbol{h}}$. Suppose a_k is picked such that

$$\frac{\mathscr{C}_k^{\boldsymbol{f}}}{\mathscr{L}_{k+1}^{\boldsymbol{h}}\mathscr{L}_k^{\boldsymbol{f}}} \leq \operatorname{sign}(\dot{\boldsymbol{h}}_{k+1}\dot{\boldsymbol{f}}_k)a_{k+1} < \frac{\mathscr{C}_k^{\boldsymbol{f}}+1}{\mathscr{L}_{k+1}^{\boldsymbol{h}}\mathscr{L}_k^{\boldsymbol{f}}}, \tag{3.84}$$

where $\operatorname{sign}(\cdot)$ *is positive/negative if the argument square matrix is positive/negative definite. Then we have an asymptotic bound*

$$\limsup_{k\to\infty} \mathbb{E}(\|\hat{\theta}_k - \theta_k^*\|^2) \leq \limsup_k \frac{a_{k+1}^2 \operatorname{tr}(\boldsymbol{R}_k) + (|a_{k+1}|\,\mathscr{L}_{k+1}^{\boldsymbol{h}} + (-1)^{\operatorname{sign}(\dot{\boldsymbol{h}}_{k+1}\dot{\boldsymbol{f}}_k)})^2 \operatorname{tr}(\boldsymbol{Q}_k)}{1 - (|a_{k+1}|\,\mathscr{L}_{k+1}^{\boldsymbol{h}}\mathscr{L}_k^{\boldsymbol{f}} - \mathscr{C}_k^{\boldsymbol{f}})^2}. \tag{3.85}$$

Proof. Notice that

$$
\begin{aligned}
&\hat{\theta}_{k+1} - \theta^*_{k+1} \\
&= \boldsymbol{f}_k(\hat{\theta}_k) + a_{k+1}[\boldsymbol{z}_{k+1} - \boldsymbol{h}_{k+1}(\boldsymbol{f}_k(\hat{\theta}_k))] - \theta^*_{k+1} \\
&= \boldsymbol{f}_k(\hat{\theta}_k) + a_{k+1}[\boldsymbol{h}_{k+1}(\theta^*_{k+1}) + \boldsymbol{v}_{k+1} - \boldsymbol{h}_{k+1}(\boldsymbol{f}_k(\hat{\theta}_k))] - (\boldsymbol{f}_k(\theta^*_k) + \boldsymbol{w}_k) \\
&= [\boldsymbol{f}_k(\hat{\theta}_k) - \boldsymbol{f}_k(\theta^*_k)] + a_{k+1}[\boldsymbol{h}_{k+1}(\boldsymbol{f}_k(\theta^*_k) + \boldsymbol{w}_k) - \boldsymbol{h}_{k+1}(\boldsymbol{f}_k(\hat{\theta}_k))] + a_{k+1}\boldsymbol{v}_{k+1} - \boldsymbol{w}_k \\
&= \left.\frac{d\boldsymbol{f}_k(\theta)}{d\theta}\right|_{\theta=\eta_1\hat{\theta}_k+(1-\eta_1)\theta^*_k} (\hat{\theta}_k - \theta^*_k) + a_{k+1}\boldsymbol{v}_{k+1} - \boldsymbol{w}_k \\
&\quad + a_{k+1}\left\{\left.\frac{d\boldsymbol{h}_{k+1}(\boldsymbol{x})}{d\boldsymbol{x}}\right|_{\boldsymbol{x}=\eta_2(\boldsymbol{f}_k(\theta^*_k)+\boldsymbol{w}_k)+(1-\eta_2)\boldsymbol{f}_k(\hat{\theta}_k)} \left[\boldsymbol{w}_k - \left(\boldsymbol{f}_k(\hat{\theta}_k) - \boldsymbol{f}_k(\theta^*_k)\right)\right]\right\} \\
&= \left.\frac{d\boldsymbol{f}_k(\theta)}{d\theta}\right|_{\theta=\eta_1\hat{\theta}_k+(1-\eta_1)\theta^*_k} (\hat{\theta}_k - \theta^*_k) + a_{k+1}\boldsymbol{v}_{k+1} - \boldsymbol{w}_k \\
&\quad + a_{k+1}\left[\left.\frac{d\boldsymbol{h}_{k+1}(\boldsymbol{x})}{d\boldsymbol{x}}\right|_{\boldsymbol{x}=\eta_2(\boldsymbol{f}_k(\theta^*_k)+\boldsymbol{w}_k)+(1-\eta_2)\boldsymbol{f}_k(\hat{\theta}_k)} \left(\boldsymbol{w}_k - \left.\frac{d\boldsymbol{f}_k}{d\theta}\right|_{\theta=\eta_3\hat{\theta}_k+(1-\eta_3)\theta^*_k} (\hat{\theta}_k - \theta^*_k)\right)\right] \\
&\equiv \left(\left.\frac{d\boldsymbol{f}_k(\theta)}{d\theta}\right|_{\theta=\tilde{\theta}_1} - a_{k+1}\left.\frac{d\boldsymbol{h}_{k+1}(\boldsymbol{x})}{d\boldsymbol{x}}\right|_{\boldsymbol{x}=\tilde{\theta}_2}\left.\frac{d\boldsymbol{f}_k}{d\theta}\right|_{\theta=\tilde{\theta}_3}\right)(\hat{\theta}_k - \theta^*_k) \\
&\quad + a_{k+1}\boldsymbol{v}_{k+1} - \left(\boldsymbol{I} - a_{k+1}\left.\frac{d\boldsymbol{f}_k}{d\theta}\right|_{\theta=\tilde{\theta}_3}\right)\boldsymbol{w}_k
\end{aligned}
\tag{3.86}
$$

where the second equation uses both (3.80) and (3.81), and the fourth equation uses mean-value theorem and assumption (d). In the last line, we use $\tilde{\theta}_1 \equiv \eta_1\hat{\theta}_k + (1-\eta_1)\theta^*_k$, $\tilde{\theta}_2 \equiv \eta_2(\boldsymbol{f}_k(\theta^*_k) + \boldsymbol{w}_k) + (1-\eta_2)\boldsymbol{f}_k(\hat{\theta}_k)$, and $\tilde{\theta}_3 \equiv \eta_3\hat{\theta}_k + (1-\eta_3)\theta^*_k$. Let us square (3.86) and taking expectations over the randomness in θ^*_0, $\boldsymbol{w}_0$,$\cdots$,$\boldsymbol{w}_k$, $\hat{\theta}_0$, $\boldsymbol{v}_1$,$\cdots$,$\boldsymbol{v}_{k+1}$, we

have the following

$$
\begin{aligned}
&\mathbb{E}(\|\hat{\theta}_{k+1} - \theta^*_{k+1}\|^2) \\
&= \mathbb{E}\left(\| \left(\left. \frac{d\boldsymbol{f}_k(\theta)}{d\theta} \right|_{\theta=\tilde{\theta}_1} - a_{k+1} \left. \frac{d\boldsymbol{h}_{k+1}(\boldsymbol{x})}{d\boldsymbol{x}} \right|_{\boldsymbol{x}=\tilde{\theta}_2} \left. \frac{d\boldsymbol{f}_k}{d\theta} \right|_{\theta=\tilde{\theta}_3} \right) (\hat{\theta}_k - \theta^*_k) \|^2 \right) \\
&\quad + a^2_{k+1} \mathrm{tr}(\boldsymbol{R}_k) + \mathbb{E}\left(\| \left(\boldsymbol{I} - a_{k+1} \left. \frac{d\boldsymbol{h}_{k+1}(\boldsymbol{x})}{d\boldsymbol{x}} \right|_{\boldsymbol{x}=\tilde{\theta}_2} \right) \boldsymbol{w}_k \|^2 \right) \\
&\leq (|a_{k+1}|\, \mathscr{L}^{\boldsymbol{h}}_{k+1} \mathscr{L}^{\boldsymbol{f}}_k - \mathscr{C}^{\boldsymbol{f}}_k)^2 \mathbb{E}(\|\hat{\theta}_k - \theta^*_k\|^2) + a^2_{k+1} \mathrm{tr}(\boldsymbol{R}_k) \\
&\quad (|a_{k+1}|\, \mathscr{L}^{\boldsymbol{h}}_{k+1} + (-1)^{\mathrm{sign}(\dot{\boldsymbol{h}}_{k+1} \dot{\boldsymbol{f}}_k)})^2 \mathrm{tr}(\boldsymbol{Q}_k). \qquad (3.87)
\end{aligned}
$$

because of assumptions (a)–(e), and the gain selection (3.84). Furthermore, the coefficient $(|a_{k+1}|\, \mathscr{L}^{\boldsymbol{h}}_{k+1} \mathscr{L}^{\boldsymbol{f}}_k - \mathscr{C}^{\boldsymbol{f}}_k)^2$ is guaranteed to be in $[0, 1)$ when (3.84) holds. Now following the derivation immediately after equation (3.34) in the proof for Theorem 3.3.1, we can obtain the asymptotic bound (3.85). □

3.5 Brief Summary

Note that in time-varying scenarios as in Section 3.1, the concentration result, instead of the improbable convergence, is the best we can hope for: $\|\hat{\theta}_k - \theta^*_k\|$ can be made small in certain statistical sense as k gets large, where θ^*_k is the time-varying target and $\hat{\theta}_k$ is the corresponding SA estimate. Under the model assumptions listed in Section 3.2, i.e., the observational noise level $\mathscr{M}_k$ in A.1, the strong convexity parameter $\mathscr{C}_k$ in A.2, the Lipschitz continuity $\mathscr{L}_k$ in A.3, and the expected drift magnitude $\mathscr{B}_k$ in A.4 are *known*,

we may implement Algorithm 1. The tracking performance of non-diminishing gain SA algorithms is guaranteed by a computable bound on MAD/RMS presented in (3.31) and in (3.36), which is useful in the analysis of finite-sample performance. The practical aspects of the finite-sample error bound for the recursion (2.1) is listed below.

- The restrictions placed on the model of the time-varying parameter is lenient compared to other assumed form of state equation. The only imposed assumption is that the average distance between two consecutive underlying parameters is strictly bounded from above. This modest assumption does not eliminate jumps in the target, and also allows the target to vary stochastically.

- A.1 allows $\hat{\boldsymbol{g}}_k(\theta)$ to be a biased estimator of $\boldsymbol{g}_k(\theta)$. Therefore, our discussion embraces many SA algorithms, including the special case of the SGD algorithm (2.1) where $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ is substituted by (2.5) discussed in [123] and SPSA in [101].

- The gain selection strategy in Lemma 3.3.6 or Algorithm 1 may provide some guidance in real-world gain-tuning. Moreover, the MAD/RMS bound informs us that the gain a_k can be neither too large nor too small—this contrasts with most prior works that claim the tracking error can be made smaller by decreasing the constant stepsize a. This is intuitive, as the ability to track time variations in θ_k^* is lost if the step-size is made too small.

- Both error bounds are computable as long as we have access to the noise level, the drift level, and the Hessian of the underlying loss function. The case of interest

requires the strong convexity of the time-varying loss function (sequence), but our tracking error bound is favorably informative under reasonable assumptions on the evolution of the true parameter being estimated. Note that our quantification for tracking capability within finite-iterations of the non-diminishing gain SA algorithm in terms of a computable error bound, can also apply to the general nonlinear SA literature. These two characteristics make our discussion different from [34, 117].

- To the best of our knowledge, there are no existing approaches in estimation theory that produce a sequence of estimates for a time-varying minimization/root-finding problem, under only the A.4 without any further stringent state evolution assumption.

In a nutshell, the iterate $\hat{\theta}_k$ provides an estimate of the optimum point θ_k^* with a certain accuracy, and the tracking errors of using $\hat{\theta}_k$ as an estimate for θ_k^* is stable for all time.

Chapter 4

Concentration Behaviors

Chapter 3 develops a MAD/RMS bound for SA algorithm (2.1) with non-decaying gain on the basis that the sampling frequency is bounded from above; i.e., the actual time elapsed between two consecutive samples, $(\tau_{k+1} - \tau_k)$, is strictly bounded away from zero for all $k \in \mathbb{N}$. This chapter will utilize the weak convergence argument (reviewed in Section 2.4) to analyze the continuous-time interpolation of SA iterates as the gain sequence approaches zero, and correspondingly $(\tau_{k+1} - \tau_k)$ goes to zero at the same order of rate. The requirement that the sampling frequency (the number of samples per unit time) has to grow as the gain sequence decreases is needed to closely follow the perpetually varying target. Even though we analyze the weak convergence limit as the gain sequence goes to zero and the number of samples per unit time grows inversely proportional to the gain sequence, the gain sequence needs not to go to zero in the actual implementation.

Many prior work on weak convergence is developed on the basis that certain averages[1] of the dynamics $\boldsymbol{g}_k(\cdot)$, denoted by $\bar{\boldsymbol{g}}(\cdot)$, do not depend on time. This assumption is appropriate if the observed data is a stationary process that evolves on a time scale that is faster than what is implied by the gain sequence. By "faster" we mean that the gain a_k is often very small compared to the time interval at which successive sets of observations are available. For example, in astronomy, the meteorological observations may be available every few hours, while the stars in the sky, in fact, changes every few seconds. On the contrary, we consider the case where the underlying θ_k^* evolves on a time-scale that is comparable with what is implied by the gain sequence. By "comparable" we mean that the gain a_k is comparable to the time difference between two consecutive observations of the moving target such as submarines and aircraft. For example, the target submarine/aircraft changes its coordinate every few seconds, and the agent that needs to track the target also need to adjust its tracking direction every few seconds. In this scenario, there is no mismatch between the model time step (the time difference between two different values of θ_k^*) and the time interval between the observation (the time difference between two consecutive noisy observations). As a result, the mean ODE (to be defined momentarily) is indeed time-dependent.

To supplement the tracking capability results in Chapter 3, this chapter characterizes the concentration behavior of the estimates using the trajectory of a nonautonomous ODE via a weak convergence argument and develops a probabilistic bound. By "concentration"

[1]For example, for every θ, $\lim_{k\to\infty}\left|\sum_{i=k}^{k+j} a_i[\boldsymbol{g}_i(\theta)-\bar{\boldsymbol{g}}(\theta)]\right| \to \mathbf{0}$ for every $j \in \mathbb{N}$.

we mean that the recursive estimates spend a majority of time arbitrarily close to some point. Namely, with an arbitrarily high probability and a small ε, the limit process is concentrated in a ε-neighborhood of some limit set of the mean ODE, if a limit set exists. The result in Section 4.1 unveils the behavior of the estimates for a small gain sequence and a finite iteration number. Then Section 4.2 provides a computable probabilistic bound to supplement the concentration result in Section 4.1, but under more stringent assumptions.

4.1 Concentration Behavior of Constant-Gain Algorithm

This section studies the concentration behavior of the SA sequence via the properties of an ODE that represents the dynamics of the algorithm. We are to establish the following proposition: with an arbitrarily fixed (usually high) probability, for small gain, the underlying data change with time should be on a scale that is commensurate with what is determined by the gain, the iterates are concentrated in an arbitrarily small neighborhood of some limit set (if one exists) of the mean ODE.

4.1.1 Basic Setup and Truncated SA Algorithm

Consider the following constrained minimization problem:

$$\text{for each } k, \text{ find } \boldsymbol{\theta} \in \boldsymbol{\Theta} \text{ s.t. } \|\boldsymbol{g}_k(\boldsymbol{\theta})\| \text{ is minimized,} \tag{4.1}$$

where $\boldsymbol{g}_k(\cdot) : \mathbb{R}^p \mapsto \mathbb{R}^p$ is a continuous mapping for each k, $\|\cdot\|$ is the vector Euclidean norm, and $\boldsymbol{\Theta} \subset \mathbb{R}^p$ is compact. Constraints are common in daily applications due to safety or economic concerns. If the (assumed unique) root of the vector-valued function $\boldsymbol{g}_k(\cdot)$ lies within $\boldsymbol{\Theta}$ for all k, then (4.1) is equivalent to a root-finding problem. Nonetheless, this general root-finding problem handles a sequence of functions $\boldsymbol{g}_k(\cdot)$ that varies with time τ_k, whereas the R-M setting [91] deals with locating the root for a single function $\boldsymbol{g}(\cdot)$ that has no k-dependence. One common application of (4.1) is immediate by letting $\boldsymbol{g}_k(\boldsymbol{\theta}) = \partial f_k(\boldsymbol{\theta})/\partial \boldsymbol{\theta}$, where $f_k(\boldsymbol{\theta})$ is a sequence of time-varying loss functions to be minimized. In this case, the problem setup (4.1) becomes the same as Subsection 3.1.1.

Different from the unconstrained SA algorithm in Chapter 3, this section discusses the projected SA algorithm (2.6) per the problem setup (4.1).

Remark 8. Although this chapter primarily discusses the constant-gain algorithm, this subsection will define terms using non-decaying gain. This general definition is in anticipation of further discussion on the adaptive gain in Chapter 5, where $\inf_k a_k > 0$, and a_k is not necessarily constant across k.

To facilitate later discussion, we introduce a projection term $\boldsymbol{\eta}_k$ and rewrite $-\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ as $\boldsymbol{\gamma}_k$. Then the projected SA algorithm (2.6) can be rearranged as a stochastic difference equation with a small step size a_k:

$$\hat{\boldsymbol{\theta}}_{k+1} = \hat{\boldsymbol{\theta}}_k + a_k \boldsymbol{\gamma}_k + a_k \boldsymbol{\eta}_k, \tag{4.2}$$

where $a_k\boldsymbol{\eta}_k = \mathscr{P}_{\boldsymbol{\Theta}}(\hat{\boldsymbol{\theta}}_k + a_k\boldsymbol{\gamma}_k) - (\hat{\boldsymbol{\theta}}_k + a_k\boldsymbol{\gamma}_k)$. That is, if $(\hat{\boldsymbol{\theta}}_k + a_k\boldsymbol{\gamma}_k)$ is not in $\boldsymbol{\Theta}$, $a_k\boldsymbol{\eta}_k$ is the vector that takes $(\hat{\boldsymbol{\theta}}_k + a_k\boldsymbol{\gamma}_k)$ back to $\boldsymbol{\Theta}$ with the shortest Euclidean norm; otherwise, $\boldsymbol{\eta}_k = \mathbf{0}$.

4.1.2 Rewrite Projected SA Algorithm (4.2) as a Stochastic Time-Dependent Process

To examine the behavior of the sequence of estimates $\{\hat{\boldsymbol{\theta}}_k\}$, we construct a continuous-time interpolation of the discrete sequence $\{\hat{\boldsymbol{\theta}}_k\}$. A natural time scale for the interpolation is the gain sequence a_k. With appropriate interpolation, a suitably constructed sequence from the iterates in (2.6) will converge to the appropriate limit set of an ODE determined by the average dynamics. If we further impose the Lyapunov stability assumption on the mean ODE, then the SA estimates "concentrates" around the stable point (if it exists) of the corresponding ODE.

i) Define $t_k = \sum_{i=0}^{k-1} a_i$. Further, define the time-mapping function $m(t)$ over the domain $[t_0, \infty)$ as: $m(t) = k$ if $t \in [t_k, t_{k+1})$ for $k \geq 0$.

ii) Define the following time-dependent step function:

$$\boldsymbol{Z}(t, \omega) \;=\; \hat{\boldsymbol{\theta}}_{m(t)}(\omega) \quad \text{for } t \geq t_0. \tag{4.3}$$

Note that $\boldsymbol{Z}(\cdot, \omega)$ is in $D(\mathbb{R} \mapsto \mathbb{R}^p)$, the space of functions that are right-continuous and have left-limits endowed with the Skorohod topology [13, Sect. 14].

iii) For $t \geq t_0$, $\boldsymbol{\gamma}_i$ is the noisy observation of $\boldsymbol{g}_i(\hat{\theta}_i)$, and $\boldsymbol{\eta}_i$ is such that $a_i\boldsymbol{\eta}_i = \mathscr{P}_{\boldsymbol{\Theta}}(\hat{\theta}_i + a_i\boldsymbol{\gamma}_i) - (\hat{\theta}_i + a_i\boldsymbol{\gamma}_i)$. Define

$$\boldsymbol{\Gamma}(t, \omega) = \mathbb{I}_{\{t \geq t_0\}} \sum_{i=0}^{m(t)-1} a_i \boldsymbol{\gamma}_i(\omega), \tag{4.4}$$

and define $\boldsymbol{X}(\cdot, \omega)$, $\boldsymbol{B}(\cdot, \omega)$, $\boldsymbol{R}(\cdot, \omega)$ analogously to $\boldsymbol{\Gamma}(\cdot, \omega)$, but using $\boldsymbol{\xi}_k$, $\boldsymbol{\beta}_k$, $\boldsymbol{\eta}_i$ in place of $\boldsymbol{\gamma}_i$ respectively.

iv) Now we may write (4.2) equivalently as

$$\begin{aligned} \boldsymbol{Z}(t + t_k) &= \hat{\theta}_k + [\boldsymbol{\Gamma}(t + t_k) - \boldsymbol{\Gamma}(t_k)] + [\boldsymbol{R}(t + t_k) - \boldsymbol{R}(t_k)] \\ &= \hat{\theta}_k + \sum_{i=k}^{m(t_k+t)-1} a_i (\boldsymbol{\gamma}_i + \boldsymbol{\eta}_i) \quad \text{if } t \geq t_0, \end{aligned} \tag{4.5}$$

The above expression lays the foundation for constructing the continuous-time version of the generalized SA.

Often, we let $t_0 = 0$ w.l.o.g.

4.1.3 Model Assumptions

In the remaining subsections, we focus on the SA algorithm (2.6) with constant gain $a_k = a > 0$. While implementing the recursive algorithms with a constant gain a in time-varying problems, the convergence in some distributional sense as the iteration index $k \to \infty$ is the best we can hope for, see Section 2.2.

Given that this section considers the behavior of $\hat{\theta}_k$ for *different* values of the constant gain $a > 0$, a superscript (a) is included to emphasize the dependency on *different* values of the constant gain a within Section 4.1. Specifically, $\hat{\theta}_k^{(a)}$, $\boldsymbol{Z}^{(a)}(\cdot)$, $\boldsymbol{\Gamma}^{(a)}(\cdot)$, $\boldsymbol{B}^{(a)}(\cdot)$, $\boldsymbol{X}^{(a)}(\cdot)$ will be used in Section 4.1 (and in this section only) to represent $\hat{\theta}_k$, $\boldsymbol{Z}(\cdot)$, $\boldsymbol{\Gamma}(\cdot)$, $\boldsymbol{B}(\cdot)$, and $\boldsymbol{X}(\cdot)$ defined in Subsection 4.1.2. However, the initialization $\hat{\theta}_0$ should be independent of a.

Let $\mathcal{F}_t^{(a)}$ be the linear space spanned by $\{\hat{\theta}_j^{(a)}, j \leq m(t)\}$, the information available up until the discrete time index $m(t)$. Let $\mathbb{E}_t^{(a)}$ represent the expectation conditioned on $\mathcal{F}_t^{(a)}$.

*Assumption B.*1. The sequence of random variables $\{\boldsymbol{\gamma}_k^{(a)}\}$ (indexed both by time index k and by gain a) is uniformly integrable. That is, $\lim_{N\to\infty} \sup_{k,a} \mathbb{E}(\mathbb{I}_{\{\|\boldsymbol{\gamma}_k^{(a)}\| \geq N\}} \cdot \|\boldsymbol{\gamma}_k^{(a)}\|) = 0$.

*Assumption B.*2. The bias sequence satisfies $\lim_{j\to\infty} \lim_{k\to\infty} j^{-1} \sum_{i=k}^{k+j-1} \mathbb{E}_{t_k}^{(a)} \boldsymbol{\beta}_i^{(a)} = \mathbf{0}$ in expectation for all $a > 0$.

*Assumption B.*3. Assume that $(\tau_{k+1} - \tau_k) \propto a$, where τ_k was defined as the actual time corresponding to the sampling instance k in Chapter 3.

Remark 9. Here, the dependency of τ_k on a is suppressed. For a fixed value of a, the corresponding sampling frequency $\tau_{k+1} - \tau_k$ is fixed for all k.

*Assumption B.*4. The sequence of measurable functions $\{\boldsymbol{g}_k(\cdot)\}$ of argument $\theta \in \boldsymbol{\Theta} \subseteq \mathbb{R}^p$ are continuous uniformly in k. Furthermore, $\mathscr{L}$ is the smallest positive real such that $\boldsymbol{g}_k(\cdot)$ are $\mathscr{L}$-Lipschitz continuous in θ for all k. Lastly, the time variability of the sequence $\{\boldsymbol{g}_k(\cdot)\}$ is such that $\|\boldsymbol{g}_{k+1}(\theta) - \boldsymbol{g}_k(\theta)\| \propto (\tau_{k+1} - \tau_k)$ for all θ

Remark 10. We do not put the (a) dependence on $\boldsymbol{g}_k(\cdot)$, as the underlying time variability of the sequence $\{\boldsymbol{g}_k(\cdot)\}$ is *not* affected by how we implement the tracking algorithm.

Let us make a few remarks regarding the aforementioned assumptions.

- A sufficient condition for B.1 is $\sup_{k,a} \mathbb{E}\|\boldsymbol{\gamma}_k^{(a)}\|^{1+\varepsilon} < \infty$ for some $\varepsilon > 0$. The commonly-used value of ε is 1. Namely, $\sup_{k,a} \mathbb{E}\|\boldsymbol{\gamma}_k^{(a)}\|^2 < \infty$.

- In the traditional SA setup where the time variability is not pertinent, we may impose stronger assumptions on the bias term than B.2; i.e., $\boldsymbol{\beta}_k^{(a)}$ is assumed to represent a bias that is asymptotically unimportant in the sense that $\mathbb{E}\|\boldsymbol{\beta}_k^{(a)}\| \to \mathbf{0}$ as $k \to \infty$ for all a. However, as explained in a footnote in Subsection 3.2.3, for FDSA or SPSA where the $\hat{\boldsymbol{g}}_k(\boldsymbol{\theta})$ is a biased estimator for $\boldsymbol{g}_k(\boldsymbol{\theta})$, the differencing interval is not allowed to decrease to zero while applied to time-varying tracking problems. Although this prevents the estimator from being "almost unbiased," this non-diminishing bias is preferred to the otherwise slower convergence and "noisier" behavior as the variance of the effective noise is inversely proportional to the square of a differencing interval. In short, the bias term $\boldsymbol{\beta}_k^{(a)}$ here could be persistent.

- B.3 is a manifestation that the sampling frequency grows inversely proportional to the constant gain a. Namely, the number of iterates per unit time has to grow inversely proportional to the gain magnitude.

- B.4 and Corollary 2.3.2 ensures that the solution to the mean ODE (to appear in (4.14) later) exists on the entire real line.

4.1.4 Main Results

This subsection illustrates how a nonautonomous differential equation can be associated with the SA iterations $\hat{\theta}_k$ generated from the projected SA algorithm (2.6) with constant gain a. We will show that, for almost all $\omega \in \Omega$, all the sample paths $\{\boldsymbol{Z}^{(a)}(\cdot, \omega)\}$ are equi-(in fact Lipschitz-) continuous in the extended sense. Then the extended Arzelà-Ascoli Theorem 2.3.2 can be applied to extract convergent subsequences whose limits satisfy the mean ODE. The path $\boldsymbol{Z}^{(a)}(\cdot, \omega)$ will closely follow the solution to the ODE (4.13) on any finite interval, with an arbitrarily high probability (uniformly w.r.t. all initial conditions within $\boldsymbol{\Theta}$) as $a \to 0$. The limit of a pathwise convergent sequence of the process $\{\boldsymbol{Z}^{(a)}(\cdot, \omega)\}$ will satisfy the mean ODE.

Lemma 4.1.1. *Assume B.1. The interpolated sequences* $\{\boldsymbol{Z}^{(a)}(\cdot, \omega)\}$ *and* $\{\boldsymbol{R}^{(a)}(\cdot, \omega)\}$ *are tight*[2] *for all* $\omega \in \Omega$.

Proof of Lemma 4.1.1. To show the uniform boundedness (which immediately implies tightness) using the uniform-integrability assumption B.1, let us first truncate the noisy observation sequence $\{\boldsymbol{\gamma}_k(\omega)\}$. For any $N > 0$, define the truncated random variables

$$\boldsymbol{\gamma}_{k,N}^{(a)}(\omega) = \begin{cases} \boldsymbol{\gamma}_k^{(a)}(\omega), & \text{if } \|\boldsymbol{\gamma}_k^{(a)}(\omega)\| \leq N, \\ \mathbf{0}, & \text{otherwise.} \end{cases}$$

[2]Tightness of a sequence of random processes was reviewed in Subsection 2.4.2.

Fix a threshold $\varepsilon > 0$, which may be arbitrarily small. Fix a time $T > 0$, which may be arbitrarily large. For some time interval $\delta > 0$, we have:

$$\begin{aligned}
&\mathbb{E}\left(\sup_{s\leq\delta,\,0\leq t\leq T-s}\|\boldsymbol{\Gamma}^{(a)}(t+s,\omega)-\boldsymbol{\Gamma}^{(a)}(t,\omega)\|\right)\\
&\quad\leq\mathbb{E}\left(\sup_{s\leq\delta,\,0\leq t\leq T-s}\sum_{i=m(t)}^{m(t+s)-1}a\|\boldsymbol{\gamma}_i^{(a)}(\omega)\|\right)\\
&\quad\leq\mathbb{E}\left[\sup_{s\leq\delta,\,0\leq t\leq T-s}\sum_{i=m(t)}^{m(t+s)-1}\left(a\|\boldsymbol{\gamma}_{i,N}^{(a)}(\omega)\|+a\|\boldsymbol{\gamma}_i^{(a)}(\omega)-\boldsymbol{\gamma}_{i,N}^{(a)}(\omega)\|\right)\right]. \qquad (4.6)
\end{aligned}$$

where both inequalities follow from the triangle inequality. On one hand, while deeming $a\sum_{i=m(t)}^{m(t+s)-1}\|\boldsymbol{\gamma}_{i,N}^{(a)}(\omega)\|$ as a function of the time t, it can change values only at multiples of a. Furthermore, at every such point, the value can change by at most Na, which goes to zero as a decreases to zero. On the other hand, Assumption B.1 implies that $\sup_{j,a}\mathbb{E}\|\boldsymbol{\gamma}_j^{(a)}(\omega)-\boldsymbol{\gamma}_{j,N}^{(a)}(\omega)\|\to 0$ as $N\to\infty$, which further implies that

$$\begin{aligned}
&\mathbb{E}\left[\sup_{s\leq\delta,\,0\leq t\leq T-s}\sum_{i=m(t)}^{m(t+s)-1}a\|\boldsymbol{\gamma}_i^{(a)}(\omega)-\boldsymbol{\gamma}_{i,N}^{(a)}(\omega)\|\right]\\
&\quad\leq\mathbb{E}\left[\sup_{a,\,m(t)\leq i\leq m(t+\delta)-1}\left(a\cdot\frac{\delta}{a}\cdot\|\boldsymbol{\gamma}_i^{(a)}(\omega)-\boldsymbol{\gamma}_{i,N}^{(a)}(\omega)\|\right)\right]\\
&\quad\leq\delta\sup_{a,\,m(t)\leq i\leq m(t+\delta)-1}\mathbb{E}\|\boldsymbol{\gamma}_i^{(a)}(\omega)-\boldsymbol{\gamma}_{i,N}^{(a)}(\omega)\|\\
&\quad\to 0\ \text{ as } N\to\infty. \qquad (4.7)
\end{aligned}$$

Combing above observations, we know that for any given threshold $\varepsilon > 0$, there exists a finite $N(\varepsilon)$ and a finite $a(\varepsilon)$ that depend on ε, such that both terms on the r.h.s. of (4.6) are less than $\varepsilon/2$. Given the arbitrariness of $\varepsilon > 0$, we claim that the paths of $\{\boldsymbol{\Gamma}^{(a)}(\cdot, \omega)\}$ indexed by a are asymptotically continuous in t w.p.1, and

$$\lim_{\delta \to 0} \limsup_{a \to 0} \mathbb{E}\left(\sup_{s \leq \delta,\, 0 \leq t \leq T-s} \| \boldsymbol{\Gamma}^{(a)}(t+s, \omega) - \boldsymbol{\Gamma}^{(a)}(t, \omega) \| \right) = 0. \tag{4.8}$$

Moreover, the statement in (4.8) also holds if $\boldsymbol{R}^{(a)}(\cdot, \omega)$ replaces $\boldsymbol{\Gamma}^{(a)}(\cdot, \omega)$ because

$$\| \boldsymbol{R}^{(a)}(t+s, \omega) - \boldsymbol{R}^{(a)}(t, \omega) \| \leq a \sum_{i=m(t)}^{m(t+s)-1} \| \boldsymbol{\gamma}_i^{(a)}(\omega) \|. \tag{4.9}$$

Also, the statement in (4.8) also holds if $\boldsymbol{Z}^{(a)}(\cdot, \omega)$ replaces $\boldsymbol{\Gamma}^{(a)}(\cdot, \omega)$ because

$$\boldsymbol{Z}^{(a)}(t, \omega) = \hat{\boldsymbol{\theta}}_0 + \boldsymbol{\Gamma}^{(a)}(t, \omega) + \boldsymbol{R}^{(a)}(t, \omega). \tag{4.10}$$

In short, with (arbitrarily) high probability, the processes $\boldsymbol{R}^{(a)}(\cdot, \omega)$ and $\boldsymbol{Z}^{(a)}(\cdot, \omega)$ change slightly on the small time interval $[t,\, t+s] \subset [0, T]$. Therefore, the condition (2.15) for Lemma 2.4.1 in proving tightness is met for the random processes $(\boldsymbol{Z}^{(a)}(\cdot, \omega), \boldsymbol{R}^{(a)}(\cdot, \omega))$ in the functional space $D(\mathbb{R} \mapsto \mathbb{R}^{2p})$. Given that $\{\boldsymbol{\theta}_k^*\}$ is constrained within $\boldsymbol{\Theta}$ and is independent of a, and $\boldsymbol{R}^{(a)}(t_0) = \mathbf{0}$ at a valid initialization $\hat{\boldsymbol{\theta}}_0 \in \boldsymbol{\Theta}$, the condition (2.14) is also met. By Lemma 2.4.1, we claim the tightness of the sequence of random processes $\{\boldsymbol{Z}^{(a)}(\cdot, \omega), \boldsymbol{R}^{(a)}(\cdot, \omega)\}$. □

Now we proceed to extract and characterize a proper sequence of continuous-time interpolations $(\boldsymbol{Z}^{(a)}(\cdot,\omega),\boldsymbol{R}^{(a)}(\cdot,\omega))$. By Prohorov's theorem [33, p. 104], we can extract a convergent sequence $(\boldsymbol{Z}^{(a_n)}(\cdot,\omega),\boldsymbol{R}^{(a_n)}(\cdot,\omega))$ as $n\to\infty$ based on Lemma 4.1.1. Let $(\boldsymbol{Z}(\cdot,\omega),\boldsymbol{R}(\cdot,\omega))$ be the process such that on the space $D(\mathbb{R}\mapsto\mathbb{R}^{2p})$, we have

$$(\boldsymbol{Z}^{(a_n)}(\cdot,\omega),\boldsymbol{R}^{(a_n)}(\cdot,\omega)) \text{ converges weakly to } (\boldsymbol{Z}(\cdot,\omega),\boldsymbol{R}(\cdot,\omega)) \text{ as } n\to\infty. \quad (4.11)$$

Lemma 4.1.2. *Assume B.1. There exists a sequence $a_n\to 0$ as $n\to\infty$, such that for all $\omega\notin\mathcal{N}$ where $\mathcal{N}$ is a set of null measure, the weak convergence limit of $\{\boldsymbol{Z}^{(a_n)}(\cdot,\omega),\boldsymbol{R}^{(a_n)}(\cdot,\omega)\}$, which was denoted as $(\boldsymbol{Z}(\cdot,\omega),\boldsymbol{R}(\cdot,\omega))$ in (4.11), is equicontinuous in the extended sense (see Definition 2.3.2).*

Proof of Lemma 4.1.2. First of all, the weak convergence limit $(\boldsymbol{Z}(\cdot,\omega),\boldsymbol{R}(\cdot,\omega))$ defined in (4.11) have continuous paths w.p.1. thanks to Lemma 2.4.2.

We now show the equi- (in fact Lipschitz-) continuity of paths of the weak sense limit $(\boldsymbol{Z}(\cdot,\omega),\boldsymbol{R}(\cdot,\omega))$ w.p.1. From (4.8), we know that

$$\lim_{\delta\to 0}\limsup_{a\to 0}\mathbb{P}\left\{\omega:\sup_{s\le\delta,\,0\le t\le T-s}\|\boldsymbol{\Gamma}^{(a)}(t+s,\omega)-\boldsymbol{\Gamma}^{(a)}(t,\omega)\|\ge\varepsilon\right\}=0,\ \ \forall\varepsilon>0,T>0. \quad (4.12)$$

Therefore, for each $T>0$, there exists a random variable $\mathscr{L}(T,\omega)<\infty$ (which is independent of the sequence index a) such that for $0\le t\le t+s\le T$,

$\|\mathbf{\Gamma}^{(a)}(t+s,\omega)-\mathbf{\Gamma}^{(a)}(t,\omega)\| \leq \mathscr{L}(T,\omega)\, s$ w.p.1. Therefore, the sequence of random processes $\{\mathbf{\Gamma}^{(a)}(\cdot,\omega)\}$ (indexed by a) is locally Lipschitz continuous w.p.1 uniformly for all a, thanks to B.1. Given (4.9) and (4.10), both $\boldsymbol{R}^{(a)}(\cdot,\omega)$ and $\boldsymbol{Z}^{(a)}(\cdot,\omega)$ is also locally Lipschitz continuous w.p.1 uniformly across a. The result then follows.

In summary, for stochastic processes $(\boldsymbol{Z}^{(a)}(\cdot,\omega),\boldsymbol{R}^{(a)}(\cdot,\omega))$ indexed by a, there exists a sequence a_n, which goes to 0 as $n\to\infty$, such that the weak convergence limit of $(\boldsymbol{Z}^{(a_n)}(\cdot,\omega),\boldsymbol{R}^{(a_n)}(\cdot,\omega))$, denoted as $(\boldsymbol{Z}(\cdot,\omega),\boldsymbol{R}(\cdot,\omega))$, is equicontinuous in the extended sense w.p.1. □

Remark 11. Under given assumptions, we know that the weak limit of the sequence $(\boldsymbol{Z}^{(a_n)}(\cdot,\omega),\boldsymbol{R}^{(a_n)}(\cdot,\omega))$ indexed by n is equicontinuous in the extended sense w.p.1, and the error vanishes for almost all ω along that sequence a_n only.

For succinctness, we will suppress ω whenever appropriate.

Theorem 4.1.1. *Assume B.1, B.2, B.3, B.4. The weak convergence limit* $(\boldsymbol{Z}(\cdot,\omega),\boldsymbol{R}(\cdot,\omega))$ *in (4.11) satisfies the mean ODE:*

$$\dot{\theta}=-\overline{\boldsymbol{g}}(t,\theta)+\boldsymbol{r}(t),\quad \boldsymbol{r}(t)\in-\mathrm{Cone}(\boldsymbol{Z}(t)), \tag{4.13}$$

where $\boldsymbol{r}(t)$ *is the adjustment needed to keep* $\theta(t)$ *within* $\mathbf{\Theta}$*, the notion of the convex cone was introduced immediately after (2.10), and* $\overline{\boldsymbol{g}}(t,\theta)$ *is the limit of*

$$\boldsymbol{g}(t,\theta)=\sum_{k=0}^{\infty}\left[\mathbb{I}_{\{t_k\leq t<t_{k+1}\}}\cdot\left(\frac{t_{k+1}-t}{a}\boldsymbol{g}_k(\theta)+\frac{t-t_k}{a}\boldsymbol{g}_{k+1}(\theta)\right)\right] \tag{4.14}$$

as $a \to 0$.

Proof. First, from the result in Lemma 4.1.2 and Theorem 2.3.1, the process $(\boldsymbol{Z}(\cdot,\omega),\boldsymbol{R}(\cdot,\omega))$ defined in (4.11) is continuous uniformly on each bounded interval, and in fact has Lipschitz continuous paths w.p.1.

Now we characterize its limit of measures of the process $(\boldsymbol{Z}(\cdot,\omega),\boldsymbol{R}(\cdot,\omega))$ on appropriate path space such that the limit measure induces a process on the path space supported on some set of limit trajectories of the ODE (4.13). In addition to $\boldsymbol{\Gamma}^{(a)}(\cdot,\omega)$, $\boldsymbol{B}^{(a)}(\cdot,\omega)$, $\boldsymbol{X}^{(a)}(\cdot,\omega)$, and $\boldsymbol{R}^{(a)}(\cdot,\omega)$ defined in (4.4), we also define

$$\boldsymbol{G}^{(a)}(t,\omega)=\int_0^t \boldsymbol{g}(s,\boldsymbol{Z}^{(a)}(s,\omega))ds \text{ for } t\geq 0, \tag{4.15}$$

and

$$\boldsymbol{\rho}^{(a)}(t,\omega)=a\sum_{i=0}^{m(t)-1}\boldsymbol{g}_i(\hat{\theta}_i(\omega))-\int_0^t \boldsymbol{g}(s,\boldsymbol{Z}^{(a)}(s,\omega))ds \text{ for } t\geq 0.$$

By the decomposition (3.5), specifically,

$$\hat{\boldsymbol{g}}_k^{(a)}(\hat{\theta}_k^{(a)})=\boldsymbol{g}_k(\hat{\theta}_k^{(a)})+\boldsymbol{\beta}_k^{(a)}(\hat{\theta}_k^{(a)})+\boldsymbol{\xi}_k^{(a)}(\hat{\theta}_k^{(a)}), \tag{4.16}$$

and following the construction in Subsection 4.1.2, we can rewrite (4.5) as:

$$\boldsymbol{Z}^{(a)}(t)=\hat{\theta}_0-\boldsymbol{G}^{(a)}(t)-\boldsymbol{\rho}^{(a)}(t)-\boldsymbol{B}^{(a)}(t)-\boldsymbol{X}^{(a)}(t)+\boldsymbol{R}^{(a)}(t). \tag{4.17}$$

Furthermore, define the following random processes with paths in $D\left(\mathbb{R}\mapsto\mathbb{R}^p\right)$:

$$\begin{aligned}\boldsymbol{W}^{(a)}\left(t\right) &= \boldsymbol{Z}^{(a)}\left(t\right)-\hat{\boldsymbol{\theta}}_0+\boldsymbol{G}^{(a)}\left(t\right)-\boldsymbol{R}^{(a)}\left(t\right)\\ &= -\boldsymbol{X}^{(a)}\left(t\right)-\boldsymbol{B}^{(a)}\left(t\right)-\boldsymbol{\rho}^{(a)}\left(t\right).\end{aligned}\tag{4.18}$$

Our goal is to show that for each existing sequence a_n arising in Lemma 4.1.2, $\boldsymbol{W}^{(a_n)}\left(t\right)$ converges weakly to a martingale process w.r.t. $\mathcal{F}_t^{(a_n)}$ spanned by $\{\hat{\boldsymbol{\theta}}_j^{(a_n)}, j\leq m\left(t\right)\}$ as $n\to\infty$. Thanks to [65, Thm. 7.4.1 on p. 234], we only need to show that for any time $t\geq 0$ and time-interval $\delta\geq 0$, and for any integer $S>0$,

$$\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{W}^{(a_n)}\left(t+\delta\right)-\boldsymbol{W}^{(a_n)}\left(t\right)\right]\right\}\overset{n\to\infty}{\longrightarrow}\mathbf{0},\tag{4.19}$$

where $j_s\in\{0,1,\cdots,m\left(t\right)\}$ for all $1\leq s\leq S$, and $F\left(\cdot\right)$ is any bounded and continuous function that maps $\mathbb{R}^{2pS}$ to $\mathbb{R}$. Given (4.18), we only need to show the following in order to show (4.19):

$$\begin{cases}\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{X}^{(a_n)}\left(t+\delta\right)-\boldsymbol{X}^{(a_n)}\left(t\right)\right]\right\}\overset{n\to\infty}{\longrightarrow}\mathbf{0},&(4.20)\\ \mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{B}^{(a_n)}\left(t+\delta\right)-\boldsymbol{B}^{(a_n)}\left(t\right)\right]\right\}\overset{n\to\infty}{\longrightarrow}\mathbf{0},&(4.21)\\ \mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{\rho}^{(a_n)}\left(t+\delta\right)-\boldsymbol{\rho}^{(a_n)}\left(t\right)\right]\right\}\overset{n\to\infty}{\longrightarrow}\mathbf{0}.&(4.22)\end{cases}$$

Let us first show that the r.h.s. of (4.20) is $\mathbf{0}$. By construction (4.16), $\boldsymbol{\xi}_k^{(a)}=\hat{\boldsymbol{g}}_k^{(a)}(\hat{\boldsymbol{\theta}}_k^{(a)})-\mathbb{E}_{t_k}^{(a)}[\hat{\boldsymbol{g}}_k^{(a)}(\hat{\boldsymbol{\theta}}_k^{(a)})]$. Hence, $\{\boldsymbol{X}^{(a_n)}(s), s\leq t\}$ is $\mathcal{F}_t^{(a_n)}$-measurable. Furthermore, the process

$\boldsymbol{X}^{(a_n)}(t)$ is an $\mathcal{F}_t^{(a_n)}$-martingale. Then by iterated conditioning and (4.16):

$$\begin{aligned}
&\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{X}^{(a_n)}(t+\delta)-\boldsymbol{X}^{(a_n)}(t)\right]\right\}\\
&\quad=\mathbb{E}\left(\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{X}^{(a_n)}(t+\delta)-\boldsymbol{X}^{(a_n)}(t)\right]\middle|\,\mathcal{F}_t^{(a_n)}\right\}\right)\\
&\quad=\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\cdot\mathbb{E}\left[\boldsymbol{X}^{(a_n)}(t+\delta)-\boldsymbol{X}^{(a_n)}(t)\middle|\,\mathcal{F}_t^{(a_n)}\right]\right\}\\
&\quad=\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\cdot\mathbb{E}\left[a_n\sum_{i=m(t)}^{m(t+\delta)-1}\boldsymbol{\xi}_i^{(a_n)}\middle|\,\mathcal{F}_t^{(a_n)}\right]\right\}\\
&\quad=\mathbb{E}\left\{\left[F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\right]\cdot\mathbf{0}\right\}\\
&\quad=\mathbf{0}.
\end{aligned}\tag{4.23}$$

We then show that the r.h.s. of (4.21) goes to $\mathbf{0}$ as the gain $a_n\to 0$. By iterated conditioning,

$$\begin{aligned}
&\|\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{B}^{(a_n)}(t+\delta)-\boldsymbol{B}^{(a_n)}(t)\right]\right\}\|\\
&=\|\mathbb{E}\left(\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{B}^{(a_n)}(t+\delta)-\boldsymbol{B}^{(a_n)}(t)\right]\middle|\,\mathcal{F}_t^{(a_n)}\right\}\right)\|\\
&=\|\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\cdot\mathbb{E}\left[\boldsymbol{B}^{(a_n)}(t+\delta)-\boldsymbol{B}^{(a_n)}(t)\middle|\,\mathcal{F}_t^{(a_n)}\right]\right\}\|\\
&=\|\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\cdot\mathbb{E}\left[a_n\sum_{i=m(t)}^{m(t+\delta)-1}\boldsymbol{\beta}_i^{(a_n)}\middle|\,\mathcal{F}_t^{(a_n)}\right]\right\}\|\\
&\le\|\mathbb{E}\left[F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)},\cdots,\hat{\boldsymbol{\theta}}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\right]\|\cdot\|\mathbb{E}\left\{\mathbb{E}\left[a_n\sum_{i=m(t)}^{m(t+\delta)-1}\boldsymbol{\beta}_i^{(a_n)}\middle|\,\mathcal{F}_t^{(a_n)}\right]\right\}\|,
\end{aligned}\tag{4.24}$$

where the second term $\|\mathbb{E}\{\mathbb{E}[a_n \sum_{i=m(t)}^{m(t+\delta)-1} \boldsymbol{b}_i^{(a_n)} \Big| \mathcal{F}_t^{(a_n)}]\}\|$ goes to $\mathbf{0}$ as $n \to \infty$ by B.2.

Let us finally show that the r.h.s. of (4.22) goes to $\mathbf{0}$ as the gain $a_n \to 0$. Observe from our construction (4.14) and by B.4, $\boldsymbol{\rho}^{(a)}(t) \to \mathbf{0}$ uniformly in t as $a \to 0$. Consequently, $\|\boldsymbol{\rho}^{(a)}(t+\tau) - \boldsymbol{\rho}^{(a)}(t)\| \to 0$ uniformly in t for any time interval δ as $a \to 0$. Specifically,

$$
\begin{aligned}
&\|\boldsymbol{\rho}^{(a)}(t+\delta) - \boldsymbol{\rho}^{(a)}(t)\| \\
&= \|a \sum_{i=m(t)}^{m(t+\delta)-1} \boldsymbol{g}_i(\hat{\boldsymbol{\theta}}_i) - \int_t^{t+\delta} \boldsymbol{g}(s, \boldsymbol{Z}^{(a)}(s))ds\| \\
&= \|a \sum_{i=m(t)}^{m(t+\delta)-1} \boldsymbol{g}_i(\hat{\boldsymbol{\theta}}_i) - \frac{a}{2} \sum_{i=m(t)}^{m(t+\delta)-1} \Big[\boldsymbol{g}_i(\hat{\boldsymbol{\theta}}_i) + \boldsymbol{g}_{i+1}(\hat{\boldsymbol{\theta}}_i)\Big]\| \\
&= \|\frac{a}{2} \sum_{i=m(t)}^{m(t+\delta)-1} \Big[\boldsymbol{g}_i(\hat{\boldsymbol{\theta}}_i) - \boldsymbol{g}_{i+1}(\hat{\boldsymbol{\theta}}_i)\Big]\| \\
&\leq \frac{a}{2} \cdot \frac{\delta}{a} \sup_{m(t)\leq i\leq m(t+\delta)-1} \|\boldsymbol{g}_i(\hat{\boldsymbol{\theta}}_i) - \boldsymbol{g}_{i+1}(\hat{\boldsymbol{\theta}}_i)\| \\
&= \frac{\delta}{2} \sup_{m(t)\leq i\leq m(t+\delta)-1} \|\boldsymbol{g}_i(\hat{\boldsymbol{\theta}}_i) - \boldsymbol{g}_{i+1}(\hat{\boldsymbol{\theta}}_i)\|. \qquad (4.25)
\end{aligned}
$$

Then by iterated conditioning and replacing a by a_n, we have:

$$
\begin{aligned}
&\|\mathbb{E}\left\{F(\hat{\theta}_{j_1}^{(a_n)},\cdots,\hat{\theta}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{\rho}^{(a_n)}(t+\delta)-\boldsymbol{\rho}^{(a_n)}(t)\right]\right\}\| \\
&=\|\mathbb{E}\left(\mathbb{E}\left\{F(\hat{\theta}_{j_1}^{(a_n)},\cdots,\hat{\theta}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\left[\boldsymbol{\rho}^{(a_n)}(t+\delta)-\boldsymbol{\rho}^{(a_n)}(t)\right]\Big|\,\mathcal{F}_t^{(a_n)}\right\}\right)\| \\
&=\|\mathbb{E}\left\{F(\hat{\theta}_{j_1}^{(a_n)},\cdots,\hat{\theta}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\cdot\mathbb{E}\left[\boldsymbol{\rho}^{(a_n)}(t+\delta)-\boldsymbol{\rho}^{(a_n)}(t)\big|\,\mathcal{F}_t^{(a_n)}\right]\right\}\| \\
&\leq\|\mathbb{E}\left[F(\hat{\theta}_{j_1}^{(a_n)},\cdots,\hat{\theta}_{j_S}^{(a_n)},\boldsymbol{\eta}_{j_1}^{(a_n)},\cdots,\boldsymbol{\eta}_{j_S}^{(a_n)})\right]\| \\
&\quad\cdot\|\mathbb{E}\left\{\mathbb{E}\left[\boldsymbol{\rho}^{(a_n)}(t+\delta)-\boldsymbol{\rho}^{(a_n)}(t)\big|\,\mathcal{F}_t^{(a_n)}\right]\right\}\| \\
&\to\mathbf{0}, \qquad (4.26)
\end{aligned}
$$

where the last line goes to zero because of B.3, B. B.4, and above observation.

To characterize the limit process of sequence, we also define the following in parallel to (4.18):

$$\boldsymbol{W}(t)=\boldsymbol{Z}(t)-\hat{\theta}_0+\boldsymbol{G}(t)-\boldsymbol{R}(t) \qquad (4.27)$$

where $\boldsymbol{Z}(\cdot)$ and $\boldsymbol{R}(\cdot)$ are defined in (4.11), and $\boldsymbol{G}(t)$ is defined by replacing $\boldsymbol{g}(\cdot,\cdot)$ in (4.15) with $\overline{\boldsymbol{g}}(\cdot,\cdot)$. It then follows that $\boldsymbol{W}(t)$ is a function of $\{\boldsymbol{Z}(s),\boldsymbol{R}(s),s\leq t\}$. The validity of (4.20)–(4.22) gives rise to (4.19) given the decomposition in (4.18).

Note that we have shown the validity of (4.19) and that $\boldsymbol{W}^{(a_n)}(t,\omega)$ is uniformly integrable. By Theorem 2.4.1, there exists a probability space $(\widetilde{\Omega},\widetilde{\mathcal{F}},\widetilde{\mathbb{P}})$ with processes $(\widetilde{\boldsymbol{Z}}^{(a_n)}(\cdot),\widetilde{\boldsymbol{R}}^{(a_n)}(\cdot),\widetilde{\boldsymbol{W}}^{(a_n)}(\cdot))$ and $(\widetilde{\boldsymbol{Z}}(\cdot),\widetilde{\boldsymbol{R}}(\cdot),\widetilde{\boldsymbol{W}}(\cdot))$, which have the same distribution as the processes $(\boldsymbol{Z}^{(a_n)}(\cdot),\boldsymbol{R}^{(a_n)}(\cdot),\boldsymbol{W}^{(a_n)}(\cdot))$ and $(\boldsymbol{Z}(\cdot),\boldsymbol{R}(\cdot),\boldsymbol{W}(\cdot))$ on $(\Omega,\mathcal{F},\mathbb{P})$. As

the way $\hat{\boldsymbol{\theta}}_k$ and $\boldsymbol{\eta}_k$ appear in (4.3) and (4.4), $\tilde{\hat{\boldsymbol{\theta}}}_k$ and $\widetilde{\boldsymbol{\eta}}_k$ can be similarly defined on the probability space $(\widetilde{\Omega}, \widetilde{\mathcal{F}}, \widetilde{\mathbb{P}})$. Furthermore, $(\widetilde{\boldsymbol{Z}}^{(a_n)}(\cdot), \widetilde{\boldsymbol{R}}^{(a_n)}(\cdot), \widetilde{\boldsymbol{W}}^{(a_n)}(\cdot))$ converges to $(\widetilde{\boldsymbol{Z}}(\cdot), \widetilde{\boldsymbol{R}}(\cdot), \widetilde{\boldsymbol{W}}(\cdot))$ w.p.1 under the (Skorohod) topology within $D([0,\infty) \mapsto \mathbb{R}^{3p})$. Recall the statement "the convergence of a sequence of functions in $D([0,\infty) \mapsto \mathbb{R}^p)$ to a continuous function in $C([0,\infty) \mapsto \mathbb{R}^p)$ in the Skorohod topology is equivalent to convergence uniformly on each bounded time interval in $C([0,\infty) \mapsto \mathbb{R}^p)$" from Section 2.4. Therefore, we know that

$$\mathbb{E}\left\{F(\widetilde{\hat{\boldsymbol{\theta}}}_{j_1}^{(a_n)}, \cdots, \widetilde{\hat{\boldsymbol{\theta}}}_{j_S}^{(a_n)}, \widetilde{\boldsymbol{\eta}}_{j_1}^{(a_n)}, \cdots, \widetilde{\boldsymbol{\eta}}_{j_S}^{(a_n)}) \left[\widetilde{\boldsymbol{W}}(t+\delta) - \widetilde{\boldsymbol{W}}(t)\right]\right\} = \mathbf{0}, \tag{4.28}$$

where $j_s \in \{0, 1, \cdots, m(t)\}$ for all $1 \leq s \leq S$, and $F(\cdot)$ is any bounded and continuous real-valued function. Since the processes are within the expectation operator $\mathbb{E}$ in (4.28), the underlying probability space is no longer relevant. Now based upon the weak convergence (4.11) derived from Lemma 4.1.1 and the Skorohod embedding argument, we claim that $(\boldsymbol{Z}^{(a_n)}(\cdot,\omega), \boldsymbol{R}^{(a_n)}(\cdot,\omega), \boldsymbol{W}^{(a_n)}(\cdot,\omega))$ converges weakly to $(\boldsymbol{Z}(\cdot,\omega), \boldsymbol{R}(\cdot,\omega), \boldsymbol{W}(\cdot,\omega))$ as $n \to \infty$ uniformly on each interval $[0,T]$. Together with the uniform integrability of every element in the sequence $\{\boldsymbol{W}^{(a_n)}\}$, we have

$$\mathbb{E}\left\{F(\hat{\boldsymbol{\theta}}_{j_1}^{(a_n)}, \cdots, \hat{\boldsymbol{\theta}}_{j_S}^{(a_n)}, \boldsymbol{\eta}_{j_1}^{(a_n)}, \cdots, \boldsymbol{\eta}_{j_S}^{(a_n)}) \left[\boldsymbol{W}(t+\delta) - \boldsymbol{W}(t)\right]\right\} = \mathbf{0} \tag{4.29}$$

where $F(\cdot)$ is any bounded and continuous real-valued function of its arguments and $j_s \in \{0, 1, \cdots, m(t)\}$ for all $1 \leq s \leq S$. By [65, Thm. 7.4.1 on p. 234], (4.29) implies that

$\boldsymbol{W}(\cdot)$ is a martingale,

$$\mathbb{E}\left[\boldsymbol{W}(t+\delta)-\boldsymbol{W}(t)|\,\boldsymbol{Z}(s),\boldsymbol{R}(s),s\leq t\right]=\mathbf{0}. \tag{4.30}$$

Combined with the weak convergence result (4.11) derived from Lemma 4.1.1, Lemma 4.1.2 shows that both $\boldsymbol{Z}(\cdot)$ and $\boldsymbol{R}(\cdot)$ have Lipschitz continuous paths w.p.1. By [65, Thm. 4.1.1 on p. 98], (4.27) implies that $\boldsymbol{W}(\cdot)$ is a constant w.p.1. Since $\boldsymbol{W}(0)=\mathbf{0}$, we have $\boldsymbol{W}(t)=\mathbf{0}$ for all t. Ultimately, for all $\omega\notin\mathcal{N}$ where the null set $\mathcal{N}$ is specified in Lemma 4.1.2, we have

$$\boldsymbol{Z}(t,\omega)=\boldsymbol{Z}(0,\omega)-\int_0^t\overline{\boldsymbol{g}}(s,\boldsymbol{Z}(s,\omega))\,ds+\boldsymbol{R}(t,\omega) \tag{4.31}$$

Note that $\boldsymbol{R}(0,\omega)=\mathbf{0}$ and $\boldsymbol{Z}(t,\omega)\in\boldsymbol{\Theta}$ for all t. Namely, the process $\boldsymbol{R}(\cdot,\omega)$ is constructed to balance the dynamics $\overline{\boldsymbol{g}}(\cdot,\boldsymbol{Z}(\cdot,\omega))$ at each time t, so that $\boldsymbol{Z}(\cdot,\omega)$ is within $\boldsymbol{\Theta}$ for all time. Specifically, $\boldsymbol{\eta}_k=\mathbf{0}$ if $\hat{\theta}_{k+1}\in\boldsymbol{\Theta}^0$ and $\boldsymbol{\eta}_k\in-\mathrm{Cone}(\hat{\theta}_{k+1})$ if otherwise. Therefore, for $s>0$, $\|\boldsymbol{R}(t+s,\omega)-\boldsymbol{R}(t,\omega)\|\leq\int_t^{t+s}\|\overline{\boldsymbol{g}}(u,\boldsymbol{Z}(u,\omega))\|du$, and therefore $\boldsymbol{R}(\cdot,\omega)$ is Lipschitz-continuous for all $\omega\notin\mathcal{N}$. By [65, Thm. 4.3.1 on p. 109], we may write $\boldsymbol{R}(t)=\int_0^t\boldsymbol{r}(s)\,ds$ where $\boldsymbol{r}(t)\in-\mathrm{Cone}(\boldsymbol{Z}(t))$ for almost all t. □

In summary, Theorem 4.1.1 deals with the limit of the sequence of the measures induced by the processes $\boldsymbol{Z}^{(a)}(\cdot)$ on the appropriate path space, and the limit measure corresponds to a process on the path space supported on some set of limit trajectories of the ODE (4.13).

Moreover, when B.4 holds, the solution to ODE (4.13) is well-defined on the entire real line due to Corollary 2.3.2.

Theorem 4.1.1 informs us that the process $\{\hat{\boldsymbol{\theta}}_k\}$ is shown to spend nearly all of its time arbitrarily close to the the limit set $\mathcal{S}_{\boldsymbol{\Theta}} \equiv \lim_{t\to\infty} \cup_{\hat{\boldsymbol{\theta}}_0 \in \boldsymbol{\Theta}} \left\{ \boldsymbol{\theta}\left(s\right), s \geq t : \boldsymbol{\theta}\left(0\right) = \hat{\boldsymbol{\theta}}_0 \right\}$, where $\boldsymbol{\theta}(\cdot)$ is the solution to a time-dependent ODE (4.13). Unfortunately, since the driving term $\overline{\boldsymbol{g}}(t, \boldsymbol{\theta})$ depends on both t and $\boldsymbol{\theta}$, we do not have much information regarding the limit set $\mathcal{S}_{\boldsymbol{\Theta}}$.

Under the special case that $\boldsymbol{g}_k(\cdot)$ varies with time yet $\boldsymbol{\theta}_k^* = \boldsymbol{\theta}^*$ for all k as in [114], the limit set is a singleton invariant set given that the trajectories are bounded within $\boldsymbol{\Theta}$ [41]. Under this special setting, we have the following corollary of "$\hat{\boldsymbol{\theta}}_k$ will spend nearly all of its time in a small neighborhood of $\mathcal{S}_{\boldsymbol{\Theta}}$ with an arbitrarily high probability," which immediately follows from Theorem 4.1.1.

Corollary 4.1.1. *Assume B.1, B.2, B.3, B.4. Further, suppose that the limit set of the time-varying ODE (4.13) is a unique point* $\boldsymbol{\theta}^*$ *and is asymptotically stable in the sense of Liapunov (discussed in Subsection 2.3.4). Then for any* $\varepsilon > 0$*, the fraction of time that* $\boldsymbol{Z}^a(\cdot)$ *will stay within the* ε*-neighborhood of the limit set* $\{\boldsymbol{\theta}^*\}$*, on* $[0, T]$ *grows to one in probability as* $a \to 0$ *and* $T \to \infty$*. Specifically, there exist* $\varepsilon_k \to 0$*,* $T_k \to \infty$ *such that* $\lim_{k\to\infty} \mathbb{P}\left\{\sup_{t\leq T_k} \|\boldsymbol{Z}^{(a)}(t) - \boldsymbol{\theta}^*\| \geq \varepsilon_k\right\} = 0$.

Unfortunately, we cannot write out an explicit expression for the rate at which the probability $\mathbb{P}\left\{\sup_{t\leq T_k} \|\boldsymbol{Z}^{(a)}(t) - \boldsymbol{\theta}^*\| \geq \varepsilon_k\right\}$ goes to zero. This is why we impose more assumptions and develop Section 4.2.

4.2 Probabilistic Bound

The previous section shows that a proper continuation of the estimate $\{\hat{\theta}_k^{(a)}\}$ converges weakly to the trajectory of the mean ODE (4.13) as the constant gain $a \to 0$. Under the special case that the limit set of the ODE is a singleton, we have the concentration result in Corollary 4.1.1. A natural question that follows is the concentration rate. Unfortunately, we cannot determine the distribution of $(\hat{\theta}_k - \theta_k^*)/\sqrt{a}$ due to the unknown evolution of θ_k^*.

Therefore, this section instead develops a computable upper bound of the probability that $\hat{\theta}_k$ generated from constant-gain SGD algorithm deviates from the trajectory of the IVP (to appear). The constant-gain SGD recursion [106, Chap. 5] is simply replacing $\hat{\boldsymbol{g}}_k(\cdot)$ in (2.1) by $\hat{\boldsymbol{g}}^{\mathrm{SG}}(\cdot)$ in (2.5) and replacing a_k by a:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a\hat{\boldsymbol{g}}_k^{\mathrm{SG}}(\hat{\theta}_k), \tag{4.32}$$

where $\hat{\boldsymbol{g}}_k^{\mathrm{SG}}(\cdot)$ is an unbiased estimator for $\boldsymbol{g}_k(\cdot)$. The main theoretical result on the finite-horizon behavior to appear is quite similar to that in [73]:

$$\forall T < \infty, \forall \varepsilon > 0, \quad \mathbb{P}\left\{\max_{0 \le k \le T/a} \|\hat{\theta}_k - \boldsymbol{Z}(t_k)\| > \varepsilon\right\} \le C(\varepsilon, a, T), \tag{4.33}$$

where for fixed $T < \infty$, $C(\varepsilon, a, T)$ tends to zero as a tends to 0, $\boldsymbol{Z}(t)$ denotes the solution to the IVP (to be defined momentarily). This result asserts that $\{\hat{\theta}_k\}_{k \ge 0}$ is a perturbed discrete-time approximation of the nonautonomous ODE with discretization step a. The

main tool in establishing the connection is the formula for variation of parameters reviewed in Subsection 2.3.3.

Remark 12. If convenient stability assumptions (similar to Corollary 4.1.1) are satisfied by the IVP, there exists a corresponding statement for infinite T (see [27] and [6, Corr. 2 on p. 43] for further details). For succinctness, we discuss the finite-time performance with $T < \infty$ only.

We should point out that the availability of a computable probabilistic bound requires more stringent assumptions than those imposed in the previous section. Specifically, this section presents a finite-time probabilistic bound on the accuracy of the estimate (for tracking a discrete-time varying target) coming from a constant-gain SGD algorithm (4.32). [123] provides the tracking error bound, whereas the probabilistic bound presented in [124] characterizes the behavior of the estimates during the process of tracking and can be used to characterize the uncertainty via confidence regions.

4.2.1 Basic Setup

We follow the problem setup (3.1) in Chapter 3, i.e., our goal is to estimate the time-varying value(s) for θ that minimize the instantaneous scalar-valued loss function $f_k(\cdot)$. Unlike Chapter 3 that considers (2.1) in general, we consider the special case (4.32).

Consider the following IVP:

$$\begin{cases} \dfrac{d}{dt}\boldsymbol{\theta}(t) = -\boldsymbol{g}(t, \boldsymbol{\theta}), \quad t \geq t_0, \\ \boldsymbol{\theta}(t_0) = \hat{\boldsymbol{\theta}}_0, \end{cases} \tag{4.34}$$

and its *perturbed* system

$$\begin{cases} \dfrac{d}{dt}\boldsymbol{\zeta}(t) = -\left[\boldsymbol{g}(t, \boldsymbol{\zeta}) + \boldsymbol{\xi}(t, \boldsymbol{\zeta})\right], \quad t \geq t_0, \\ \boldsymbol{\zeta}(t_0) = \hat{\boldsymbol{\theta}}_0, \end{cases} \tag{4.35}$$

where both $\boldsymbol{g}(\cdot, \cdot)$ and $\boldsymbol{\xi}(\cdot, \cdot)$ are maps from $\mathbb{R} \times \mathbb{R}^p$ to $\mathbb{R}^p$.

Consider $\boldsymbol{g}(\cdot, \cdot)$ defined in (4.14). This $\boldsymbol{g}(t, \boldsymbol{\theta})$ is measurable in t and continuously differentiable in $\boldsymbol{\theta}$ with bounded derivatives uniformly w.r.t. t. The $\boldsymbol{\xi}(\cdot, \cdot)$ function can be similarly defined by substituting $\boldsymbol{\xi}_k(\cdot)$ for $\boldsymbol{g}_k(\cdot)$ in (4.14). Such $\boldsymbol{\xi}(\cdot, \cdot)$ is measurable in t and Lipschitz in $\boldsymbol{\theta}$ uniformly w.r.t. t. Now that $\boldsymbol{\xi}(t, \boldsymbol{\zeta})$ is the linear interpolation of the measurement noise $\boldsymbol{\xi}_k(\omega)$ at sample point ω (the dependence on ω is suppressed), we will analyze the behavior of $\boldsymbol{\zeta}(t)$ at each sample point, i.e., with a fixed ω, the system (4.35) is effectively deterministic at a given sample point ω.

4.2.2 Model Assumptions

Assumption B. 5. The sequence $\{\boldsymbol{\xi}_k\}_{k=0}^{K-1}$ is mutually *independent*, not necessarily identically distributed random vectors with mean $\mathbf{0}$ and bounded magnitude $\|\boldsymbol{\xi}_k\| \leq \mathscr{M}$ for all k almost surely. The value of the error does not depend on the evaluation point $\boldsymbol{\theta}$.

*Assumption B.*6. The function $\boldsymbol{g}_k$ is continuously differentiable. Furthermore, $\mathscr{C}$ is the smallest positive real such that $\boldsymbol{g}_k(\cdot)$ satisfies $(\theta - \zeta)^T(\boldsymbol{g}_k(\theta) - \boldsymbol{g}_k(\zeta)) \geq \mathscr{C}\|\theta - \zeta\|^2$ for all $\theta, \zeta \in \mathbb{R}^p$ and all k.

Denote $\boldsymbol{H}_k(\theta) = \partial \boldsymbol{g}_k(\theta)/\partial \theta^T$. The following statements regarding $\boldsymbol{g}_k$, $\boldsymbol{H}_k$, and θ_k^* should be interpreted in the a.s. sense if randomness is involved.

*Assumption B.*7. The magnitude of the (discrete-time) varying gradient function is strictly bounded: $\|\boldsymbol{g}_k(\theta)\| \leq \mathscr{G}$ for all k and θ.

Here are some implications of the assumptions.

- Under B.5, the noise term does not depend on θ at all. We may use $\xi(t)$ and $\xi(t, \theta)$ interchangeably for the rest of our discussion.

  Note that the function $\xi(t)$ is, in fact, random, since it depends on the specific sample point $\omega \in \Omega$ of the stochastic process $\{\xi_k(\omega)\}_{k\geq 0}$. Namely, only one trajectory is under consideration for deterministic $\xi(t)$, whereas the *average* performance of a collection of all possible realizations of trajectories of $\{\xi_k(\omega)\}$ has to be taken into account. For succinctness, we suppress the dependence of $\xi(t)$ on ω.

- One direct consequence of B.6 and B.7 is $\|\theta_{k+1}^* - \theta_k^*\| \leq 2\mathscr{C}^{-1}\mathscr{G}$ from (3.8) and (3.7), i.e., the change of the optimal parameter between every two consecutive discrete time instances is strictly bounded. This resembles assumption A.4.

- Under B.4, the IVP (4.34) has a unique solution over $[0, T]$ for any finite T. To see this, notice that $\boldsymbol{g}(t, \theta)$ shares a common Lipschitz constant $\mathscr{L}$ w.r.t. θ for every t.

Therefore, the existence, uniqueness, and extensibility (to $t \in \mathbb{R}$) follow immediately from Corollary 2.3.2 . Furthermore, the Lagrange stability of the solution to (4.34) follows from the Gronwall-Bellman inequality [5, Lemma 1 on p. 35].

- Under B.4 and B.5, the IVP (4.35) admits a unique solution. To see this, notice that the driving term, $-\boldsymbol{g}(t, \theta) - \xi(t, \theta)$ is piecewise continuous in t, and is Lipschitz continuous w.r.t. θ, the global existence and uniqueness follow directly from [54, Thm. 3.2 on p. 93].

4.2.3 Main Results

Let us mention one caveat before we present the main results. It is desirable to increase a for maintaining tracking momentum, whereas it is necessary to decrease a for better tracking accuracy when θ_k^* is fixed at one value. Nonetheless, the gain selection is not the central topic here; it was touched on in the previous chapter and will be further discussed in Chapter 5. We assume that the pre-determined gain a enables the SGD algorithm (4.32) with a constant gain a to keep track of the target θ_k^*. Without a carefully-tuned constant gain a, once the estimate $\hat{\theta}_k$ deviates significantly from the target θ_k^*, it is likely to lose it ever after. The following discussion is based upon the availability of a tuned gain $a > 0$.

Let us first discuss a lemma to handle the noise term later on.

Lemma 4.2.1. *Assume B.5. For an arbitrarily fixed $\delta > 0$ and finite time-horizon $T > 0$,*

$$\mathbb{P}\left\{\omega : \| \int_0^T \xi(s)\, ds \| > \delta\right\} = (p+1) \exp\left(-\frac{\delta^2}{a\mathscr{M}\left(T\mathscr{M}/2 + \delta/3\right)}\right), \tag{4.36}$$

where the r.h.s. approaches 0 *exponentially as* $a \to 0$*, and the sample point* $\omega \in \Omega$ *determines*[3] *the entire measurement noise sequence* $\{\boldsymbol{\xi}_k(\omega)\}_{k\geq 0}$.

Proof of Lemma 4.2.1. Without loss of generality, assume that $m(T)$ defined in Subsection 4.1.2 equals $T/a \equiv K$. Denote the variance statistics of the sum as:

$$\nu_K \equiv \max\left\{ \| \mathbb{V}(\boldsymbol{\xi}_0) + 2\sum_{k=1}^{K-2} \mathbb{V}(\boldsymbol{\xi}_k) + \mathbb{V}(\boldsymbol{\xi}_{K-1}) \|, \right. \\ \left. \mathbb{E}\left[\|\boldsymbol{\xi}_0\|^2 + 2\sum_{k=1}^{K-2} \|\boldsymbol{\xi}_k\|^2 + \|\boldsymbol{\xi}_{K-1}\|^2 \right] \right\} \tag{4.37}$$

Under B.5, $\| \mathbb{V}(\boldsymbol{\xi}_k) \| \leq \mathrm{tr}\left[\mathbb{E}(\boldsymbol{\xi}_k \boldsymbol{\xi}_k^T)\right] \leq \mathscr{M}^2$, and $\mathbb{E}\|\boldsymbol{\xi}_k\|^2 \leq \mathscr{M}^2$ for all $0 \leq k \leq K-1$. Therefore, $\nu_K \leq 2K\mathscr{M}^2$. By [111, Thm. 1.6.2 on p. 13], we have

$$\begin{aligned} &\mathbb{P}\left\{ \omega : \| \int_0^T \boldsymbol{\xi}(s)\, ds \| > \delta \right\} \\ &= \mathbb{P}\left\{ \omega : \frac{a}{2} \| \boldsymbol{\xi}_0(\omega) + 2\sum_{k=1}^{K-2} \boldsymbol{\xi}_k(\omega) + \boldsymbol{\xi}_{K-1}(\omega) \| > \delta \right\} \\ &\leq (p+1) \exp\left(-\frac{\delta^2}{a\mathscr{M}(T\mathscr{M}/2 + \delta/3)} \right). \end{aligned} \tag{4.38}$$

The r.h.s. of (4.38) approaches 0 exponentially as a approaches 0 for fixed δ. □

Now we present the main theorem in computing the probabilistic bound.

[3] Once a ω is picked, the entire sequence $\boldsymbol{\xi}_k$ is determined.

Theorem 4.2.1. *Assume B.4, B.5, B.6, and B.7. For an arbitrarily fixed finite $T > 0$, and for any threshold $\varepsilon > 0$,*

$$\begin{aligned}&\mathbb{P}\left\{\max_{0\le k\le T/a}\|\hat{\theta}_k - \theta\left(t_k\right)\| > \varepsilon\right\}\\ &\quad\le (p+1)\exp\left[-\frac{\left(\varepsilon e^{\mathscr{C}}/2\right)^2}{a\mathscr{M}\left(T\mathscr{M}/2 + \varepsilon e^{\mathscr{C}}/6\right)}\right],\end{aligned} \tag{4.39}$$

where the r.h.s. approaches 0 exponentially as $a \to 0$, and $\theta(t)$ is the solution to (4.34).

Proof of Theorem 4.2.1. Define the following time-dependent continuous function:

$$\breve{\boldsymbol{Z}}\left(t\right) = \sum_{k=0}^{\infty}\left\{\left[\frac{t_{k+1}-t}{a}\boldsymbol{Z}(t_k) + \frac{t-t_k}{a}\boldsymbol{Z}(t_{k+1})\right]\cdot \mathbb{I}_{\{t_k\le t<t_{k+1}\}}\right\}, \tag{4.40}$$

where $\boldsymbol{Z}(t)$ was defined in (4.3). Note that $\breve{\boldsymbol{Z}}(t)$ is the linear interpolation of $\{\hat{\theta}_k\}$ at times $t_k = ka$. We have

$$\breve{\boldsymbol{Z}}(t_{k+1}) = \hat{\theta}_{k+1} = \breve{\boldsymbol{Z}}(t_k) - a[\boldsymbol{g}(t_k, \breve{\boldsymbol{Z}}(t_k)) + \boldsymbol{\xi}(t_k, \breve{\boldsymbol{Z}}(t_k))]. \tag{4.41}$$

To establish a connection between $\breve{\boldsymbol{Z}}(t_k)$ and $\theta(t_k)$, we invoke the triangle inequality and analyze the behavior of two terms: (1) $\max_{0\le t_k\le T}\|\theta(t_k) - \zeta(t_k)\|$ where $\zeta(t)$ is the solution to (4.35), and (2) $\max_{0\le t_k\le T}\|\breve{\boldsymbol{Z}}(t_k) - \zeta(t_k)\|$.

(1) First, consider term $\max_{0\le t_k\le T}\|\theta(t_k) - \zeta(t_k)\|$. The difference between the solution to the system (4.34) and the solution to the perturbed system (4.35) can be handled by the Alekseev's formula reviewed in Subsection 2.3.3. All the necessary conditions to

invoke the Alekseev's formula are met: $\boldsymbol{g}(t, \theta)$ is continuously differentiable w.r.t. θ under B.6 and the construction (4.14), the magnitude of $\partial \boldsymbol{g}\left(t, \theta\right) / \partial \theta$ is uniformly bounded under B.4, and $\xi\left(t\right)$ does not depend on θ under B.5. Let us invoke the (uniform) bound on the norm of the fundamental matrix provided in [16, Thm. 1]:

$$\begin{aligned} & \| \theta\left(t; t_0, \hat{\theta}_0\right) - \zeta\left(t; t_0, \hat{\theta}_0\right) \| \\ & \quad \leq \| \int_{t_0}^{t} \exp\left\{ \int_{s}^{t} \left[\lambda_p \left(\frac{\partial\left[-\boldsymbol{g}\left(s, \theta\right)\right]}{\partial \theta^T} \right) \right] \right\} \cdot \xi\left(s\right) ds \| \\ & \quad \leq e^{-\mathscr{C}} \| \int_{t_0}^{t} \xi\left(s, \zeta(s; t_0, \hat{\theta}_0)\right) ds \|, \end{aligned} \tag{4.42}$$

where the second inequality uses B.6 and $\lambda_{\max}\{-\partial[\boldsymbol{g}(t, \theta)]/\partial \theta^T\} \leq -\mathscr{C}$. The notions $\theta(t; t_0, \hat{\theta}_0)$ and $\zeta(t; t_0, \hat{\theta}_0)$ are to emphasize the dependence of the initialization of $\hat{\theta}_0$ at t_0 in Alekseev's formula reviewed in Subsection 2.3.3. Besides, B.4 implies $\lambda_{\min}\{-\partial[\boldsymbol{g}(t, \theta)]/\partial \theta^T\} \geq -\mathscr{L}$. Therefore, for arbitrarily given threshold $\varepsilon > 0$,

$$\mathbb{P}\left\{ \max_{0 \leq t_k \leq T} \| \theta\left(t_k\right) - \zeta\left(t_k\right) \| > \varepsilon \right\} \leq \mathbb{P}\left\{ e^{-\mathscr{C}} \| \int_{0}^{T} \xi\left(s\right) ds \| > \varepsilon \right\}. \tag{4.43}$$

(2) Now consider the term $\max_{0 \leq t_k \leq T} \| \breve{\boldsymbol{Z}}\left(t_k\right) - \zeta\left(t_k\right) \|$. [22, Thm. 212A] shows that $\max_{0 \leq t_k \leq T} \| \breve{\boldsymbol{Z}}\left(t_k\right) - \zeta\left(t_k\right) \|$ is bounded from above by $O\left(a\right)$ with a bounded constant term. Therefore, the difference between the linear interpolation of the noisy

discretization $\hat{\boldsymbol{\theta}}_{k+1} = \hat{\boldsymbol{\theta}}_k - a\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ and the perturbed system $\dot{\boldsymbol{\zeta}}(t) = -\boldsymbol{g}(t, \boldsymbol{\zeta}) - \boldsymbol{\xi}(t, \boldsymbol{\zeta})$ diminishes to zero as the discretization interval a approaches 0.

Any sample point ω in the intersection of the event $\{\omega : \max_{0 \le t_k \le T} \|\boldsymbol{\theta}(t_k, \omega) - \boldsymbol{\zeta}(t_k, \omega)\| < \varepsilon/2\}$ and $\left\{\omega : \max_{0 \le t_k \le T} \|\boldsymbol{\zeta}(t_k, \omega) - \breve{\boldsymbol{Z}}(t_k, \omega)\| < \varepsilon/2\right\}$ must fall within the event $\left\{\omega : \max_{0 \le t_k \le T} \|\breve{\boldsymbol{Z}}(t_k, \omega) - \boldsymbol{\theta}(t_k, \omega)\| < \varepsilon\right\}$. Part (2) establishes that $\max_{0 \le t_k \le T} \|\boldsymbol{\zeta}(t_k) - \breve{\boldsymbol{Z}}(t_k)\| < \varepsilon/2$ is valid almost surely as long as the constant gain a is smaller than a certain threshold specified in [22, Thm. 212A]. Combined, for *certain* gain $a > 0$ satisfying this condition, the probability that $\hat{\boldsymbol{\theta}}_k$ deviates from $\boldsymbol{\theta}(ka)$ is bounded from above by

$$C(a, \varepsilon, T) = (p+1) \exp\left[-\frac{\left(\varepsilon e^{\mathscr{C}}/2\right)^2}{a\mathscr{M}\left(T\mathscr{M}/2 + \varepsilon e^{\mathscr{C}}/6\right)}\right].$$

□

4.2.4 Further Remarks

This section analyzes the recursive iterates via the solution to an IVP. Some subtleties are worth mentioning. In the classical setting of decaying gain and fixed underlying parameter to be identified, the stationary point of the limiting autonomous ODE is shown to be the limit point of general SA algorithms under certain conditions [106, Sect. 4.3]. However, for constant-gain algorithm designed to minimize a time-varying objective

function $f_k(\cdot)$, it is not justified to transfer the terminologies, such as the concept of equilibrium, from an autonomous ODE to nonautonomous ODE (where the forcing term has explicit dependence on time). The recursive estimates never settle if the underlying parameter is perpetually time-varying. Many prior works on tracking problems assume that the time-varying objective function $f_k(\cdot)$ and its gradient function, evaluated at the values within the allowable region, have fixed limiting values[4]. Such assumption essentially forces the slowly time-varying parameter to converge to a limit for large k, and the limit point of recursive estimates will eventually coincide with the equilibrium of limiting autonomous ODE. However, this condition may not fit practical scenarios. Also note that this section focuses on the SGD algorithm (4.32), where direct unbiased measurement of the unknown gradient is available.

4.2.5 One Quick Example

This subsection provides a synthetic study in tracking a jump process to illustrate the effects of the noise, the drift, and the gain on the tracking capability.

We aim to track a jump process $\boldsymbol{\theta}_k^*$. For every k, $\boldsymbol{\theta}_{k+1}^*$ remains the same as $\boldsymbol{\theta}_k^*$ with a probability of 0.9995, and $\boldsymbol{\theta}_{k+1}^* = \boldsymbol{\theta}_k^* + \boldsymbol{v}_k$ with a probability of 0.0005, where $\boldsymbol{v}_k$ is independent and uniformly distributed on a spherical disc, with a radius of $\mathscr{G}$, centered at the origin. The time-varying loss function is $f_k(\boldsymbol{\theta}) = \mathbb{E}\|\boldsymbol{\theta} - \boldsymbol{\theta}_k^*\|^2/2$, and the

[4]For a continuously differentiable function that has a limit as $t \to \infty$, i.e., $f(t) \overset{t\to\infty}{\longrightarrow} \ell$, it is *not* necessarily the case that $f'(t) \overset{t\to\infty}{\longrightarrow} 0$. However, if a function $f \in C^1(\mathbb{R})$ satisfies both $f(t) \overset{t\to\infty}{\longrightarrow} \ell$ and $f'(t) \overset{t\to\infty}{\longrightarrow} \ell'$, then we can safely conclude that $\ell' = 0$.

corresponding gradient function is $\boldsymbol{g}_k(\theta) = \theta - \mathbb{E}\theta_k^*$ per discussion on [106, p. 70]. However, the accessible information is the noisy gradient evaluation $\hat{\boldsymbol{g}}_k(\theta) = \boldsymbol{g}_k(\theta) + \xi_k$, where ξ_k follows a truncated normal distribution with mean 0, positive definite matrix Σ, and truncation bounds $[l, u]$ on each component of $\boldsymbol{e}_k$. This distribution satisfies B.5.

To illustrate, we pick $p = 2$, $\hat{\theta}_0 = \theta_0^* = \mathbf{0}$, $a = 0.1$, $T = 5000$, $\mathscr{G} = 50$, $\Sigma = \sigma^2 \boldsymbol{I}$ with $\sigma = 1$, and the truncated normal with $l = -3$ and $u = 3$. Figure 4.1 is the scatter plot of a single realization of $\hat{\theta}_k$ and the underlying jump process θ_k^*. It is visually obvious that the estimates $\hat{\theta}_k$ are capable of tracking the time-varying jump process θ_k^*. In terms of the tracking speed and accuracy, when a jump in the $\{\theta_k^*\}$ sequence occurs, it takes *at most* 29 iterations for the estimates $\hat{\theta}_k$ to fall within the ball with a radius of 2 and a center of the newest value of θ^*.

We also run $100,000$ replicates for $a = 0.1$, and the empirical probability is listed in Figure 4.2. The empirical probability for the event $\max_{0 \le t_k \le T} \| \hat{\theta}_k - \theta(t_k) \| > 4$ happening is 0.88. Any $\varepsilon \ge 7$ gives an empirical probability of zero. Note that the magnitude of ε is still small compared to the possible jump magnitude $\mathscr{G} = 50$. This phase transition (the probability is either very close to one or very close to zero) may be attributed to these two main reasons: (1) there is a certain (unknown) stability region for the constant gain a and (2) the probability bound in Theorem 4.2.1 is *not uniformly tight* for all ε. Overall, the trajectory of $\boldsymbol{Z}(t)$, the linear continuation of $\hat{\theta}_k$, can be characterized by the trajectory of $\theta(t)$, the solution to IVP (4.34).

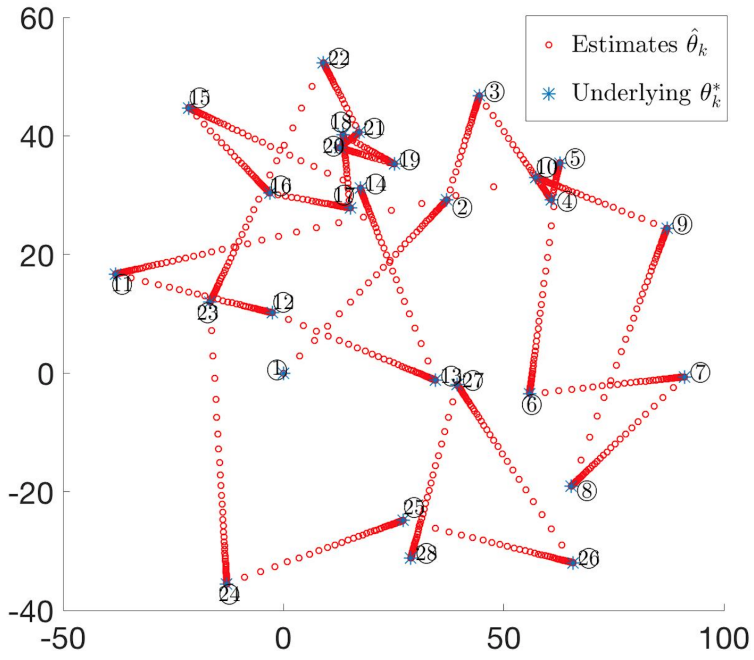


Figure 4.1: The underlying time-varying jump process Θ_k^* and $\breve{\boldsymbol{Z}}(t)$ generated by $\hat{\Theta}_k$, with $a = 0.1$. For all k, we have $\breve{\boldsymbol{Z}}(t_k) = \hat{\Theta}_k$. The number in the circles corresponds to the counter of the jumps.

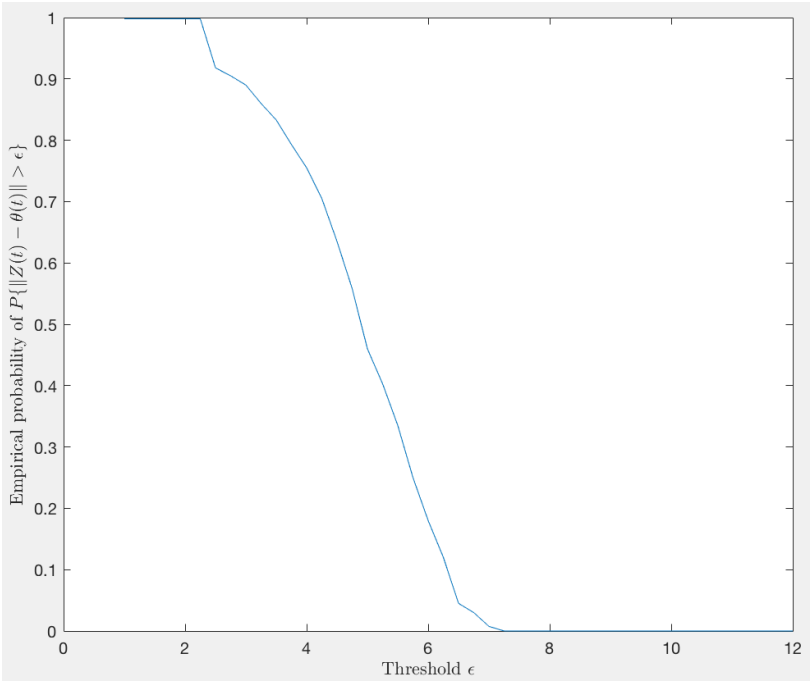


Figure 4.2: The empirical probability that $\boldsymbol{Z}(t)$ deviates from $\Theta(t)$ by at least ε as a function of ε. Note that $\boldsymbol{Z}(t_k) = \hat{\Theta}_k, \forall k$.

4.3 Concluding Remarks

Our work investigates a class of stochastic approximation algorithms that allows for time-varying loss functions and nonlinear dynamics. In the nonstationary scenario, we cannot expect "convergence" for the constant-gain algorithm, due to a combination of observation noise and nonnegative gain. The best we can hope for is to get into a neighborhood of the optimizer (sequence). The practical implication of the weak convergence result and the probabilistic bound for SA-like tracking algorithms are listed below.

- Our framework does not require an explicit model for the time variations of θ_k^* because they are typically unknown in reality. Instead, we "bury" the variations of θ_k^* in either B.4 or B.7 as applied to $\boldsymbol{g}_k(\cdot)$. The analysis of the time-varying framework is more challenging, as the classical SA techniques cannot be applied.

- The time-varying assumption imposed on the loss function f_k is useful and necessary when the underlying system is time-varying, when successive iterations are performed on different components of the independent variable (e.g., the alternating minimization procedure), or when the experimental procedure varies with k, or when specific variance reduction methods (e.g. stratified sampling) are employed, and so on.

- Many prior works impose assumptions on the noise process and the time-varying sequences so that the dynamics "average out" to a function that does not depend on

time. However, this is rarely the case in applications. In our case, the mean ODE can be time-dependent (nonautonomous).

- The result in Theorem 4.1.1 and Corollary 4.1.1 informs us that, the smaller the step-size, the better $\hat{\theta}_k$ approximates the trajectory of $\theta(t)$. However, with the smaller a, the number of steps to simulate the time-varying ODE on $[0, T]$ with fixed T grows as T/a.

- Theorem 4.2.1 characterizes the probabilistic behavior of the recursive SGD estimates over a finite-time period. Realistically, we cannot achieve many asymptotic (as $k \to \infty$) properties of the recursive estimates, as all algorithms have to stop within finite time.

- To guarantee tracking stability, there exists an upper-bound on the gain sequence. Chapter 3 informs us that, when the sampling frequency is fixed, there exists a lower-bound on the gain sequence for tracking capability and robustness consideration.

  As in the gain-selection guidance conveyed in the previous chapter, this chapter also informs us that the gain sequence for tracking perpetually varying target should be neither too large nor too small. The trajectory of $\theta(t)$, which is the solution to the ODE (4.13) or (4.34), does not coincide with the true θ_k^* sequence at every τ_k, although they are close, see [116].

We should mention that in the results (3.32), (3.37), and (3.41) back in Chapter 3, the limit is taken over the iteration number k given the adaptive gain selected according to Algorithm 1. Here, the result in Theorem 4.2.1 is valid for the entire time-frame, and the maximization is taken over the iteration number given a fixed constant gain a. The probabilistic bound in Theorem 4.2.1 provides a general sense of the likelihood of $\hat{\theta}_k$ staying close to θ_k^* for a constant gain a under Assumptions B.4, B.5, B.6, and B.7. Besides, in Theorem 4.1.1 here, the weak convergence limit is taken over the constant gain a. It should be interpreted that for some nonzero constant gain a, which needs not go to zero, the continuous interpolation of the estimates $\hat{\theta}_k$ will stay "close" (in the sense of weak limit) to the ODE (4.13) under the conditions therein and when the underlying data should change with time at a rate that is commensurate with what is determined by the gain.

Chapter 5

Data-Dependent Gain-Tuning

In time-varying SA problems, the gains in the recursive schemes must be strictly bounded away from zero to accommodate the time variability in the target values $\{\theta_k^*\}$. This characteristic distinctively differs from the classical SA algorithms with diminishing gains that place lesser weights on more recent information. In general, the SA algorithms with non-decaying gain a_k, such as Algorithm 1 in Chapter 3, are capable of tracking time-varying targets. Nonetheless, the optimal value of a_k depends on the knowledge of the drift $(\theta_{k+1}^* - \theta_k^*)$, which we do not know. Therefore, we have to provide an estimate for the step-size a_k on top of the estimation of θ_k^* in Chapter 3. Often, a constant gain, $a_k = a$ for all k, is used in (2.1), for both the ease of implementation and the consequent tracking algorithm robustness. It has been observed that the constant-gain SGD algorithm (4.32) is capable of tracking a time-varying target under certain conditions [69]. However,

the tracking performance is rather sensitive to the constant gain a, and gain tuning remains an unsettled practical issue [64] [6, p. 160].

Recall that the gain selection strategy in Algorithm 1 requires knowledge of both the Lipschitz constant $\mathscr{L}_k$ and the convexity parameter $\mathscr{C}_k$, which may be unknown in practical applications. Though it may be possible to estimate these parameters by collecting multiple observations at each time instant k, such "multiple sequential measurements at a time" implementation is contradictory with the general SA philosophy[1] of "averaging across iterations" and the time-varying setting. It is also prohibitive due to the computational overhead (and possibly equipment cost) within each iteration. Instead, we consider a more restrictive time-varying scenario summarized in C.3, C.5, or C.6 (to appear). With more stringent assumptions, we can detect regime change using Algorithm 2, adapting the gain sequence correspondingly using Algorithm 3. The main advantages here are that we do not require $\mathscr{L}_k$ and $\mathscr{C}_k$ to be known in advance for gain-tuning purposes.

5.1 Detecting Jumps/Changes

This section proposes a method for jump/change detection. We consider a special case summarized in Assumption C.3 to appear, which is motivated by the hybrid systems mentioned in Subsection 2.2.1. Hybrid systems are routinely modeled by a finite number of *diffusion*s with different drift and diffusion coefficients, and a random *jump* process

[1] In history, there were attempts to approximate $f(\theta)$ by averaging several i.i.d. measurements of $y(\theta)$. However, this approach turns out to be theoretically inefficient and numerically prohibitive. The cost of obtaining noisy measurements used to approximate $f(\theta)$ at a single point could have been allocated to help to minimize $f(\theta)$—after all, minimization is the primary objective.

modulates these diffusions with a known transition matrix. Since the diffusion and the jump structures are rarely available to the experimenter, we set aside the diffusion component and abstract the jump component via C.3. To keep it simple, we consider the constant-gain SGD algorithm (4.32), where the gain a requires advance tuning.

5.1.1 Basic Change Detection Setup

In a typical change-detection setup, we receive a sequence of observations $\mathbf{x}_1, \mathbf{x}_2, \cdots$, which are realizations of a sequence of random variables $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots$. Several number of abrupt change points $\kappa_1, \kappa_2, \cdots$ divide the sequence of random variables into segments, where the observations within each segment are i.i.d.[2]. That is,

$$\boldsymbol{x}_i \sim \begin{cases} F_0, & \text{if } i \leq \kappa_1, \\ F_1, & \text{if } \kappa_1 + 1 \leq i \leq \kappa_2, \\ F_2, & \text{if } \kappa_2 + 1 \leq i \leq \kappa_3, \\ \vdots \end{cases} \tag{5.1}$$

for some set of distributions $\{F_0, F_1, \cdots\}$, and that $F_i \neq F_{i+1}$ for all i. The goal of this section is to estimate the set of change points $\{\kappa_i\}$.

[2] Although the assumption of independent observation between change points may seem restrictive, this is not the case since a statistical model can usually be fitted to the observations to model any dependence, with change detection then being performed on the independent residuals [45].

Detection Criteria

The performance of online change detection algorithms is typically measured by two criteria [4]. Take the situation where the length of observations is fixed at K and there is only one possible change point κ for example. The first criterion is the "average run length," $\mathrm{ARL}_0 \equiv \mathbb{E}\left(\hat{\kappa} | \, \boldsymbol{x}_i \sim F_0 \text{ for } i \leq K\right)$, which is defined as the average number of observations until a changepoint is detected, when the algorithm is run over a sequence of observations with no changepoints (i.e., false positive). A false positive is said to have occurred if $\hat{\kappa} < \kappa$. The second criterion is the "mean detection delay," $\mathrm{ARL}_1 \equiv \mathbb{E}\left(\hat{\kappa} - \kappa | \, \boldsymbol{x}_i \sim F_0 \text{ for } i \leq \kappa \text{ and } \boldsymbol{x}_i \sim F_1 \text{ for } \kappa < i \leq K\right)$, is defined as the average number of observations between a changepoint occurring and the change being detected (i.e., a mean delay). In general, an acceptable value of ARL_0 is chosen before attempting to minimize the detection delay. This is analogous to the Neyman-Pearson testing setup, where a Type-II error is minimized subject to the Type-I error being bounded from above.

Relation With Control Chart and Change Detection

Note that a great deal of difficulty in our setting comes from that we only have access to $\{\hat{\theta}_k\}$ instead of $\{\theta_k^*\}$ itself.

Connection. The challenge of the online monitoring involves a sequence of changes of unknown and varying magnitude at unknown time instances. Furthermore, there is

no universal criterion for accessing the detection performance in an online monitoring framework.

Distinction. A majority of the change detection literature assumes direct access, though it may be noisy, of the underlying process θ_k^*. However, we only get to access $\hat{\theta}_k$, whose explicit distributional relation with the time-varying θ_k^* is unknown. As a consequence, our proposed change detection strategy inevitably has lower power compared to the scenario where we can observe θ_k^* directly.

5.1.2 Model Assumptions

We consider a simplified scenario for the hybrid diffusions mentioned at the end of Sect. 2.2.1. Initially, the jump process rests at one of its states/regimes, denoted as $\theta_{(\mathsf{s})}^*$, and the continuous component evolves per the diffusion process (with associated drift and diffusion). Then after a random duration of time $[\text{start}_\mathsf{s}, \text{end}_\mathsf{s}]$, a jump occurs. The discrete process then switches to a new state $\theta_{(\mathsf{s}+1)}^*$, and, accordingly, the diffusion process changes its drift and diffusion matrix within another random duration of time $[\text{start}_{\mathsf{s}+1}, \text{end}_{\mathsf{s}+1}]$ with $\text{start}_{\mathsf{s}+1} = \text{end}_\mathsf{s} + 1$. The jump component will remain in this state/regime until the next jump, and the diffusion/oscillation component will not change its drift or diffusion matrix until a new jump takes place, and so on.

*Assumption C.*1 (Regime-specified quadratic function form). For $k \in [\text{start}_\mathsf{s}, \text{end}_\mathsf{s}]$ with $\mathsf{s} \in \mathbb{Z}$, the loss function takes the form of $f_k(\theta) = (\theta - \theta_{(\mathsf{s})}^*)^T \boldsymbol{H}_{(\mathsf{s})}(\theta - \theta_{(\mathsf{s})}^*)/2$, and the

gradient takes the form of $\boldsymbol{g}_k(\theta) = \boldsymbol{H}_{(\mathsf{s})}(\theta - \theta^*_{(\mathsf{s})})$ for some symmetric and positive-definite matrix $\boldsymbol{H}_{(\mathsf{s})}$.

*Assumption C.*2 (Error ξ_k is zero-mean and bounded-variance). For $k \in [\text{start}_\mathsf{s}, \text{end}_\mathsf{s}]$ the sequence $\{\xi_k\}$ is i.i.d. with mean $\mathbf{0}$ and a bounded covariance matrix of $\boldsymbol{V}_{(\mathsf{s})}$. That is, the observation noise ξ_k in (2.5) depends on the state/regime *only*.

*Assumption C.*3 (Abstraction of jump component). The lengths $(\text{end}_\mathsf{s} - \text{start}_\mathsf{s})$ of the random durations for $\theta^*_k = \theta^*_{(\mathsf{s})}$ are i.i.d. with geometric distribution having a mean of $\mathcal{J}^{-1}$, where $\mathcal{J}$ is the jump/change probability (usually less than 5%). Furthermore, assume that $(\text{end}_\mathsf{s} - \text{start}_\mathsf{s}) > \mathsf{w} \equiv \max\{p+1, \mathcal{J}^{-1}/10\}$ w.p.1.

*Assumption C.*4 (Abstraction of general trend-stationary system, including both jump and diffusion components). In addition to C.3, let $\mathscr{B}'$ be the smallest number such that the within-regime oscillation is restricted by $\|\theta^*_k - \theta^*_{(\mathsf{s})}\| \leq \mathscr{B}'$ w.p.1. for $\text{start}_\mathsf{s} \leq k \leq \text{end}_\mathsf{s}$ and for all $\mathsf{s} \in \mathbb{N}$. Further assume that $\mathscr{B}'$ is no larger than $\mathscr{B}$, where $\mathscr{B}$ is the smallest number such that the cross-regimes jump is restricted by $\|\theta^*_{(\mathsf{s}+1)} - \theta^*_{(\mathsf{s})}\| \leq \mathscr{B}$.

Let us also provide some remarks regarding these assumptions.

- In a majority of applications to identification and adaptive system theory [70, 115], a positive-definite matrix $\boldsymbol{H}$ such that $\boldsymbol{g}(\theta) = \boldsymbol{H}(\theta - \theta^*)$ does exist; i.e., C.1 holds.

  Following the discussion in Subsection 3.2.6, denote $\mathscr{C}_{(\mathsf{s})} \equiv \lambda_{\min}(\boldsymbol{H}_{(\mathsf{s})})$ and $\mathscr{L}_{(\mathsf{s})} \equiv \lambda_{\max}(\boldsymbol{H}_{(\mathsf{s})})$.

- Given that $\mathbb{E}(\boldsymbol{\xi}_k) = \mathbf{0}$ and $\mathbb{V}(\boldsymbol{\xi}_k) = \boldsymbol{V}_{(\mathsf{s})}$ for all $k \in [\text{start}_\mathsf{s}, \text{end}_\mathsf{s}]$ as in C.2, we have $\mathbb{E}(\|\boldsymbol{\xi}_k\|^2) = \text{tr}(\boldsymbol{V}_{(\mathsf{s})})$. In fact, C.2 is an abstraction for "the diffusion/oscillation component will not change its drift or diffusion matrix until a new jump takes place."

- C.1 can be relaxed to [83, Assumption B(ii)], i.e., $\boldsymbol{g}_k(\boldsymbol{\theta}) = \boldsymbol{H}_{(\mathsf{s})}(\boldsymbol{\theta} - \boldsymbol{\theta}^*_{(\mathsf{s})}) + O(\|(\boldsymbol{\theta} - \boldsymbol{\theta}^*_{(\mathsf{s})})\|^2)$. C.2 can be relaxed to [83, Assumption A(iii)]; i.e., $\|\boldsymbol{\xi}_k(\hat{\boldsymbol{\theta}}_k)\|^2$ can be upper bounded by a linear function of $\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*_k\|$ w.p.1. We use stronger assumptions in order to present the results more elegantly.

- C.3 captures the jump part of the "diffusion and jump and so on and so forth" nature of the hybrid system, and discards the oscillation part for the time being. C.3 does not require that the number of states is finite, as long as the jump probability is small. Even though our detection algorithm (summarized in Algorithm 2 to appear) is developed based on C.3, our numerical result supports that it is also robust to the case where the following C.4 holds. C.4 is less stringent than C.3 and captures both the oscillation and the jump components for hybrid systems.

- The assumption $\text{end}_\mathsf{s} - \text{start}_\mathsf{s} > \mathsf{w}$ is imposed so that the duration of each regime $\boldsymbol{\theta}^*_{(\mathsf{s})}$ is sufficiently long such that

  (i) the normal approximation [83] of constant-gain estimates takes effect;

  (ii) the full rank of our pooled variance estimate (5.7) to appear is ensured;

  (iii) a sufficient amount of data can be gathered to compute the needed statistics (5.8) to appear for the p-dimensional problem.

(iv) When C.3 holds, we say that a (regime) change arises at time $(k+1)$ if θ^*_{k+1} differs from θ^*_k. When C.4 holds, we say that a change arises at time $(k+1)$ if $\theta^*_k \in \mathrm{Ball}_{\mathscr{B}'}(\theta^*_{(\mathsf{s})})$ and $\theta^*_{k+1} \in \mathrm{Ball}_{\mathscr{B}'}(\theta^*_{(\mathsf{s}+1)})$.

When $\mathsf{end}_\mathsf{s} - \mathsf{start}_\mathsf{s}$ is sufficiently long, depending on the starting value $\hat{\theta}_{\mathsf{start}_\mathsf{s}}$, the process $\hat{\theta}_k$ given by (4.32) may first show a phase of steadily approaching the solution $\theta^*_{(\mathsf{s})}$, and then shows the oscillation around $\theta^*_{(\mathsf{s})}$ without further approaching $\theta^*_{(\mathsf{s})}$. We will call them the *transient phase* (known as search phase in [84]) and the *steady-state phase* (known as stationary/convergence phase in [84]) throughout the rest of our discussion.

5.1.3 Base Case: One *Unknown* Change Point Occurs For $\kappa \leq K$

Let us start with a simplified scenario where $\theta^*_i = \theta^*_\mathsf{A}$ for $1 \leq i \leq \kappa$ and $\theta^*_j = \theta^*_\mathsf{B}$ for $\kappa < j \leq K$, such that there is a *single* hypothesized change point at time $1 < \kappa < K$. At each time instant k, we test the null hypothesis

$$\mathtt{H}_0 : \theta^*_1 = \cdots = \theta^*_K, \tag{5.2}$$

versus the alternative hypothesis

$$\mathtt{H}_1 : \theta^*_1 = \cdots = \theta^*_k, \theta^*_k \neq \theta^*_{k+1}, \theta^*_{k+1} = \cdots = \theta^*_K \text{ for some } k \text{ such that } 2 \leq k \leq K-2.$$

During the first regime $k \leq \kappa$, let $\boldsymbol{g}_k(\theta)$ be $\boldsymbol{H}_{\mathsf{A}}(\theta - \theta^*_{\mathsf{A}})$ and $\mathbb{V}(\xi_k) = \boldsymbol{V}_{\mathsf{A}}$. During the second regime $\kappa < k \leq K$, let $\boldsymbol{g}_k(\theta)$ be $\boldsymbol{H}_{\mathsf{B}}(\theta - \theta^*_{\mathsf{B}})$ and $\mathbb{V}(\xi_k) = \boldsymbol{V}_{\mathsf{B}}$. [84] shows that if the gain is held to a constant a, constant-gain SA estimate behaves differently compared to decaying-gain SA estimates, in that the estimates ultimately converge to a region of radius $O(\sqrt{a})$ that contains θ^* and then oscillates in that region without further approaching θ^*. The steady-state covariance of $\hat{\theta}_k$ is—for a small value of a—approximately equal to $a\boldsymbol{\Sigma}$ where $\boldsymbol{\Sigma}$ is the solution of $\boldsymbol{H}\boldsymbol{\Sigma} + \boldsymbol{\Sigma}\boldsymbol{H} = \boldsymbol{V}$ and can be given by [112]:

$$\boldsymbol{\Sigma} = \int_0^\infty e^{t\boldsymbol{H}} \boldsymbol{V} e^{t\boldsymbol{H}^T} dt, \quad \text{or,} \quad \operatorname{vec}(\boldsymbol{\Sigma}) = (\boldsymbol{I} \otimes \boldsymbol{H} + \boldsymbol{H}^T \otimes \boldsymbol{I})^{-1} \operatorname{vec}(\boldsymbol{V}). \tag{5.3}$$

When Assumption C.3 holds, we expect that $\hat{\theta}_k$ will quickly reach the steady-state phase within each regime after a short period of a transient phase provided that the gain a is pre-tuned carefully. Immediately, $a \cdot \operatorname{tr}(\boldsymbol{\Sigma})$ is approximately equal to $\mathbb{E}(\|\hat{\theta}_k - \theta^*\|^2)$ during the steady-state phase. Hence, for $k \leq \kappa$, we expect $\hat{\theta}_k$ to be *approximately* normally distributed with a mean of θ^*_{A} and a variance matrix $a\boldsymbol{\Sigma}_{\mathsf{A}}$ given by $\boldsymbol{H}_{\mathsf{A}}\boldsymbol{\Sigma}_{\mathsf{A}} + \boldsymbol{\Sigma}_{\mathsf{A}}\boldsymbol{H}_{\mathsf{A}} = \boldsymbol{V}_{\mathsf{A}}$. Let us ignore the transient behavior after the jump point κ for the time being. We expect that, for $\kappa < k \leq K$, $\hat{\theta}_k$ is going to be approximately normally distributed with a mean of θ^*_{B} and a variance matrix $a\boldsymbol{\Sigma}_{\mathsf{B}}$ given by $\boldsymbol{H}_{\mathsf{B}}\boldsymbol{\Sigma}_{\mathsf{B}} + \boldsymbol{\Sigma}_{\mathsf{B}}\boldsymbol{H}_{\mathsf{B}} = \boldsymbol{V}_{\mathsf{B}}$. Of course, neither $\boldsymbol{H}_{\mathsf{A}}$ ($\boldsymbol{H}_{\mathsf{B}}$) nor $\boldsymbol{V}_{\mathsf{A}}$ ($\boldsymbol{V}_{\mathsf{B}}$) is known in reality. Since the information $\boldsymbol{V}_{\mathsf{A}}$ ($\boldsymbol{V}_{\mathsf{B}}$) and $\boldsymbol{H}_{\mathsf{A}}$ ($\boldsymbol{H}_{\mathsf{B}}$) needed to construct $\boldsymbol{\Sigma}_{\mathsf{A}}$ ($\boldsymbol{\Sigma}_{\mathsf{B}}$) are *not* revealed to the agent(s), we can *not* take full advantage of the multivariate-normal approximation or to detect the regime states,

Test Statistic

The *multivariate Behrens–Fisher problem* deals with testing the equality of means from two multivariate normal distributions when the dispersion matrices are *unknown* and potentially *unequal*. It inherits all the difficulties arising in the univariate Behrens–Fisher problem, including estimating the dispersion matrix using data, and the distributional approximation. Define:

$$\bar{\theta}_{i:j} \;=\; \frac{1}{j-i+1}\sum_{l=i}^{j}\hat{\theta}_l, \quad \text{for } i \le j, \tag{5.4}$$

and

$$\boldsymbol{W}_{1:k} \;=\; \frac{1}{k\,(k-1)}\sum_{i=1}^{k}\left[(\hat{\theta}_i-\bar{\theta}_{1:k})(\hat{\theta}_i-\bar{\theta}_{1:k})^T\right], \tag{5.5}$$

$$\boldsymbol{W}_{k+1:K} \;=\; \frac{1}{(K-k)(K-k-1)}\sum_{i=k+1}^{K}\left[(\hat{\theta}_i-\bar{\theta}_{k+1:K})(\hat{\theta}_i-\bar{\theta}_{k+1:K})^T\right], \tag{5.6}$$

$$\boldsymbol{W}_k \;=\; W_{1:k}+W_{k+1:K}. \tag{5.7}$$

When Assumption C.3 holds, $\boldsymbol{W}_k$ is ensured to have full rank. To investigate a possible jump/change occurring after observation k, we use the following statistic for testing a difference between pre-change and post-change data at an *assumed* change point k as:

$$T^2_{1:k:K} \equiv \left(\bar{\theta}_{1:k}-\bar{\theta}_{k+1:K}\right)^T \boldsymbol{W}_k^{-1}\left(\bar{\theta}_{1:k}-\bar{\theta}_{k+1:K}\right), \quad k=2,\cdots,K-2. \tag{5.8}$$

Remark 13. Though (5.8) shares some similarities with the Hotelling T^2 statistic, it is fundamentally different in that the Hotelling T^2 statistic is *not* robust to unequal covariance matrices.

Distribution of Test Statistics

The main issue in applying (5.8) to detecting change for streaming data in an online fashion is that, the probability of rejecting the null via the T^2 test statistic defined in (5.8) depends on the *unknown* dispersion matrices $\mathbf{\Sigma}_{\mathsf{A}}$ and $\mathbf{\Sigma}_{\mathsf{B}}$ under the null hypothesis (5.2) that θ^*_{A} equals θ^*_{B} as in (5.2). In practice, this dependency compromises the statistical inference when the underlying true dispersion matrices $\mathbf{\Sigma}_{\mathsf{A}}$ and $\mathbf{\Sigma}_{\mathsf{B}}$ significantly deviate from each other or when the sample size is not sufficiently large to estimate them accurately. Below are some existing remedies.

The first remedy is to use $T^2_{1:k:K}$ with an approximation of its degrees of freedom [119]:

$$
\begin{aligned}
&T^2_{1:k:K} \sim \frac{\nu_{1:k:K} p}{\nu_{1:k:K} - p + 1} F_{p, \nu_{1:k:K} - p + 1}, \\
&\text{with } \nu_{1:k:K} = \left\{ \frac{1}{k} \left[\frac{\boldsymbol{d}_k^T \boldsymbol{W}_k^{-1} \boldsymbol{W}_{1:k} \boldsymbol{W}_k^{-1} \boldsymbol{d}_k}{\boldsymbol{d}_k^T \boldsymbol{W}_k^{-1} \boldsymbol{d}_k} \right]^2 + \frac{1}{K-k} \left[\frac{\boldsymbol{d}_k^T \boldsymbol{W}_k^{-1} \boldsymbol{W}_{k+1:K} \boldsymbol{W}_k^{-1} \boldsymbol{d}_k}{\boldsymbol{d}_k^T \boldsymbol{W}_k^{-1} \boldsymbol{d}_k} \right]^2 \right\}^{-1},
\end{aligned}
\tag{5.9}
$$

where $F_{\cdot,\cdot}$ (with two positive inputs) denotes the probability distribution function for F-distribution with given degrees of freedoms, $\boldsymbol{d}_k = \bar{\theta}_{1:k} - \bar{\theta}_{k+1:K}$. In addition to the

approximation in (5.9), there are several others, including Johansen's approximation [50], and Nel and Van der Merwe's approximation [78].

The second remedy follows from [59] which proposed another approximation where the approximated degrees of freedom is guaranteed to be nonnegative:

$$T^2_{1:k:K} \sim \frac{\nu_{1:k:K} p}{\nu_{1:k:K} - p + 1} F_{p,\nu_{1:k:K}-p+1},$$
$$\text{with } \nu_{1:k:K} = \frac{p + p^2}{*}, \tag{5.10}$$

where the $*$ in (5.10) is

$$* = \frac{1}{k-1}\left\{\text{tr}\left[(\boldsymbol{W}_{1:k}\boldsymbol{W}_k^{-1})^2\right] + \left[\text{tr}\left(\boldsymbol{W}_{1:k}\boldsymbol{W}_k^{-1}\right)\right]^2\right\}$$
$$+ \frac{1}{K-k-1}\left\{\text{tr}\left[(\boldsymbol{W}_{k+1:K}\boldsymbol{W}_k^{-1})^2\right] + \left[\text{tr}\left(\boldsymbol{W}_{k+1:K}\boldsymbol{W}_k^{-1}\right)\right]^2\right\}.$$

The approximation in (5.10) has the best known size and power since 2004.

A Change Detection Strategy for Base Case

If the change point were known a priori to be at κ, then T^2_κ will be the generalized likelihood ratio test statistic for testing a change between pre-κ and post-κ data. If the change point is unknown in advance, the maximum over all possible split points, $\max_{2\le k\le K-2} T^2_k$, is the generalized likelihood ratio test statistic for change in the mean. The maximizing index $\hat{\kappa} = \arg\max_{2\le k\le K-2} T^2_k$ is the maximum likelihood estimate of the change/jump point. Now that there is a *single* assumed change point up until time index K,

a natural estimate for $2 \leq k \leq K-2$ is

$$\hat{\kappa} = \arg \max_{2 \leq k \leq K-2} T^2_{1:k:K}. \tag{5.11}$$

This statistic fits well for a *single* change point in a *fixed* sample of size K. Unfortunately, even for such a simplified base case, we are not able to accurately provide the ARL_0 and ARL_1 for strategy (5.11), because the distribution of the maximum over a range of $T^2_{1:k:K}$ statistics gets extremely complicated.

All we can conclude is that, after $\hat{\kappa}$ is computed, we may use the *approximated* P-value, denoted as $h_{1:\hat{\tau}:K}$, to serve as a proxy for the probability of *incorrectly* announcing a change arises, whereas, in fact, no change occurs. Specifically, $h_{1:\hat{\kappa}:K}$ is calculated by $F_{p,\nu_{\hat{\kappa}},-p+1}$ evaluated at $(\nu_{\hat{\kappa}} - p + 1)T^2_{1:\hat{\tau}:K} / (\nu_{\hat{\kappa}} p)$,

$$h_{1:\hat{\kappa}:K} \equiv \int_{\frac{\nu_{\hat{\kappa}}-p+1}{\nu_{\hat{\kappa}} p} T^2_{1:\hat{\kappa}:K}}^{\infty} \frac{\Gamma\left[\frac{\nu_{\hat{\kappa}}+1}{2}\right]}{\Gamma\left(\frac{p}{2}\right)\Gamma\left(\frac{\nu_{\hat{\kappa}}-p+1}{2}\right)} \left(\frac{p}{\nu_{\hat{\kappa}}-p+1}\right)^{\frac{p}{2}} \frac{x^{\frac{p-2}{2}}}{\left[1+\left(\frac{p}{\nu_{\hat{\kappa}}-p+1}\right)x\right]^{\frac{\nu_{\hat{\kappa}}+1}{2}}} dx. \tag{5.12}$$

5.1.4 Building Block: Multiple *Unknown* Change Points For the Data Stream

When we need to detect *multiple unknown* change points for a data *stream*, the problem gets even more unwieldy. First, we do *not* know how many change points are upcoming beforehand. Second, the detection has to be performed on a stream of data $\{\hat{\theta}_k\}$ in an *online* fashion, which causes excessive storage and computational overhead for active monitoring as the stream gets longer and longer. Even for the base case where there is *one* unknown

change point, the naive strategy of computing $T^2_{1:j:k}$ for every $j < k-1$ whenever a new $\hat{\theta}_k$ comes in is unrealistic. The computational burden becomes increasingly heavy, as the datastream grows larger and larger and as the number of possible change points increases. It is again unrealistic to achieve the change detection goal promptly, i.e., correctly announce k to be a change point immediately after observing the information up till time τ_k, not to mention that the probabilistic error for $\hat{\kappa}_{s+1}$ hinges upon that for $\hat{\kappa}_s$.

To avoid further complications, we impose C.3 for the following reasons.

(i) It is assumed that the period of each regime should be sufficiently long, so that we can gradually accrue confidence in making detection decisions within a certain time-frame.

(ii) The random duration $(\text{end}_s - \text{start}_s)$ is assumed to be bounded from below by $w > 0$ w.p.1. Then at each time instant $k \geq 2w$, we can use a *fixed* amount of data, $\{\hat{\theta}_{k-2w+1}, \hat{\theta}_{k-2w+2}, \cdots, \hat{\theta}_{k-1}, \hat{\theta}_k\}$, to test whether a change arose at time index $k - w$.

The "elbow" (i.e., the hazard rate at this point is lower than its two adjacent time points) point on the curve of P-value defined in 5.12 gets identified as a change point. See the details summarized in Algorithm 2.

Remark 14. By fixing the window w, Algorithm 2 has *constant* computational complexity and a *fixed* amount of memory.

Algorithm 2 Change Detection: Constant-gain SGD Algorithm (4.32) Using *One*-Measurement at a Time

Input: a window size w (based on dimension p and $\mathcal{J}$), a constant gain a, and a P-value threshold α.

set $\hat{\theta}_0$, the best approximation available at hand to estimate θ_0^*.
for $1 \leq k < 2\mathsf{w}$ **do**
    update $\hat{\theta}_k \leftarrow \hat{\theta}_{k-1} - a\hat{\boldsymbol{g}}_{k-1}^{\mathrm{SG}}(\hat{\theta}_{k-1})$.
end for
for $k \geq 2\mathsf{w}$ **do**
    compute $\bar{\theta}_{k-2\mathsf{w}+1:k-\mathsf{w}}$ and $\bar{\theta}_{k-\mathsf{w}+1:k}$ per (5.4).
    compute $\boldsymbol{W}_{k-2\mathsf{w}+1:k-\mathsf{w}}$, $\boldsymbol{W}_{k-\mathsf{w}+1:k}$ per (5.5) to (5.5).
    compute $T_{k-2\mathsf{w}+1:k-\mathsf{w}:k}$ per (5.8).
    compute corresponding P-value $h_k \leftarrow h_{k-2\mathsf{w}+1:k-\mathsf{w}:k}$ per (5.12).
    if $k > 2\mathsf{w}+1$ **then**
        if $h_{k-1} < \min\{\alpha, h_{k-2}, h_k\}$ **then**
Output: $k-1$ as a change point.
        end if
    end if
end for

Proposition 2. *Under C.2, C.1, and C.3, the probability of incorrectly detecting that a change happened when, in fact, no change did occur, is approximately* $h_{\hat{\kappa}-\mathsf{w}+1:\hat{\tau}:\hat{\kappa}+\mathsf{w}}$*, where the function* h *(which takes three inputs) is defined in (5.12), and* $\hat{\kappa}$ *(suppressing the numbering if there are multiple identified points) is identified by Algorithm 2.*

Proposition 2 follows directly from the distribution of the test statistic (5.8) in the multivariate Behrens–Fisher problem, and the approximation hinges upon the imposed assumptions: the Hessian matrix of $f_k(\cdot)$ remains constant within each regime under C.1; the observation errors are i.i.d. mean zero within each regime under C.2. Moreover, under C.3, after ignoring the transient phase between regimes, we assume that the estimates $\hat{\theta}_k$ are approximately normally distributed around $\theta_{(\mathsf{s})}^*$ where s is such that $\mathrm{start}_{\mathsf{s}} \leq k \leq \mathrm{end}_{\mathsf{s}}$.

5.1.5 An Example for Detecting Regime Change

Let us consider an example similar to Subsubsection 3.3.2, yet different in the sense that we no longer have access to $\mathscr{C}_k$, $\mathscr{L}_k$, $\mathscr{M}_k$ and $\mathscr{B}_k$ as defined in Section 3.3. Again, consider a simple case with $p = 2$, where the (unknown) nonstationary drift evolves according to:

$$\theta^*_{k+1} = \begin{cases} \theta^*_k, & \text{with a probability of } 99.9\%, \\ \theta^*_k + 500\left(\cos(\varphi_k),\ \sin(\varphi_k)\right)^T, & \text{with a probability of } 0.1\%, \end{cases} \tag{5.13}$$

with $\theta^*_0 = \mathbf{0}$ and $\varphi_k \overset{\text{i.i.d.}}{\sim} \text{Uniform}[0, 2\pi]$. The observation error ξ_k is again i.i.d. $\text{Normal}(\mathbf{0}, \sigma_1^2 \boldsymbol{I}_p)$, and the Hessian matrix is again given in (3.42). Here, we use a constant gain $a = 1/30$, which is the inverse of the Lipschitz continuity parameter of the gradient.

Following Algorithm 2, we pick the window size w to be 25, as we are expecting a jump to arise every $1/(.1\%) = 1000$ iterations on average and the dimension $p = 2$. Figure 5.1 shows the *true* jump point, at time 241, 2412, and 4644 in red circles, and the identified jump point (a very successful identification in this case) in enlarged black stars.

As it turns out, even though the detection algorithm listed in Algorithm 2 is proposed based on C.3 where only the "jump" structure is captured, it is robust to the case where C.4 is met. Figure 5.2 below shows how Algorithm 2 detects the jump points 605, 1051, 2189,

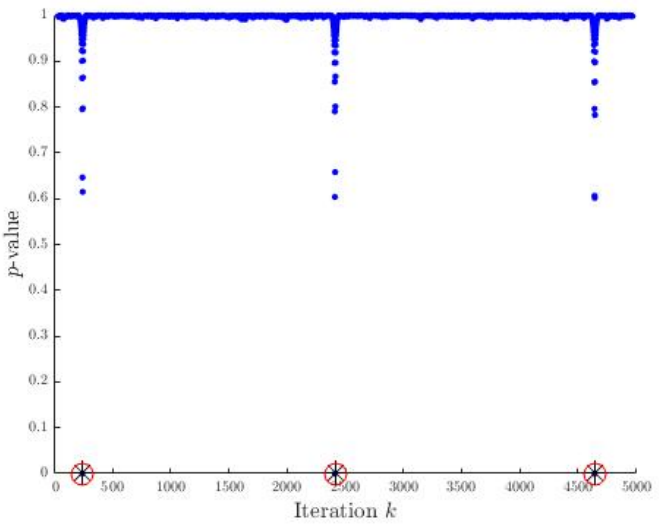


Figure 5.1: Change detection using the "elbow"-point of the P-value curve, when $\{\theta_k^*\}$ evolves according to (5.13)

3300, and 4522 when the (unknown) nonstationary drift is evolved according to:

$$\theta_k^* \text{ is i.i.d. uniformly distributed within } \{\theta | \, \|\theta - \theta_{(\mathsf{s})}^*\| \leq 50\}, \text{ for } \text{start}_\mathsf{s} \leq k \leq \text{end}_\mathsf{s}, \tag{5.14}$$

where the jump probability of the sequence $\{\theta_{(\mathsf{s})}^*\}$ is again 0.1%.

5.1.6 Further Remarks

It is natural to envision the adaptation of a distribution-free, nonparametric test statistic, whose distribution under the null is *independent* of the data to streamline the above change detection procedure and to produce a desired false alarm rate (FAR) to be maintained for any stream. Granted, [56, 67, 92] discussed several univariate distribution-free test statistics, aiming to detect a change in the location and/or scale parameter of a stream

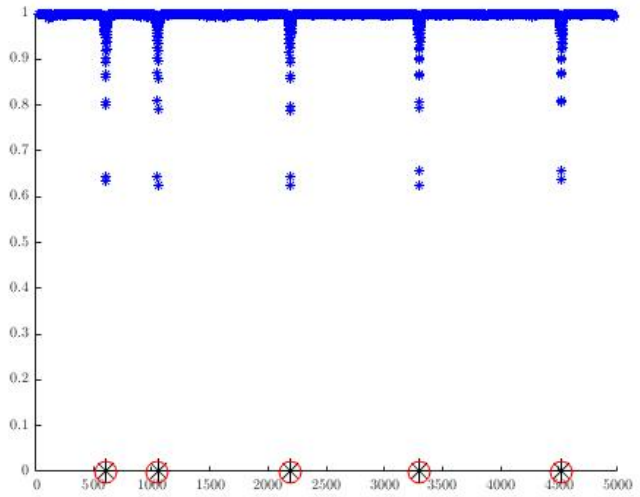


Figure 5.2: Change detection using the "elbow"-point of the P-value curve, when $\{\theta_k^*\}$ evolves according to (5.14)

of random variables. However, we do not consider the nonparametric method due to the expensive computational cost of computing ranks or the depth of the minimum spanning tree for the multivariate extension [38] in an online fashion and the low power of a general nonparametric test that uses a distribution-free test statistic.

This section discusses a strategy to detect the jump components in the time-varying sequence $\{\theta_k^*\}$ by making use of the constant-gain SGD estimates $\hat{\theta}_k$. Nonetheless, Algorithm 2 still requires keeping track of the last $2w$ SGD estimates, and can only identify the jump after w iterations, even though it has a favorable detection power. The upcoming section discusses a way to identify jumps *instantly* by imposing more stringent assumptions than C.3.

5.2 Gain Adaptation

We still focus on the constant-gain recursion (4.32) and propose a gain-adaptation strategy using the available information $\mathcal{F}_{k+1}$ at time τ_k. In general, there is no guarantee that the constant-gain SGD estimates will converge to a fixed θ^*. Under weak conditions, [28] shows that constant-gain SGD estimate $\hat{\theta}_k$ exhibits positive variance (uniformly bounded away from zero) for all k. Moreover, [28] also proves that the constant-gain SGD iterates converge to their unique stationary distribution exponentially fast in k. The results of [28] are consistent with practical experience that the constant-gain SA makes rapid progress in approaching θ^*, yet it remains in the neighborhood of θ^* afterward. Therefore, we are motivated to perform the following:

a) increase the stepsize for faster cross-regime adaptation once a jump is detected;

b) reduce the stepsize in a controlled manner to further reach the vicinity of $\theta^*_{(s)}$ once the iterates are determined to oscillate around $\theta^*_{(s)}$.

Different from the previous section where Algorithm 2 announces a change arises after w observations after the change point, this section aims to make the announcement as *soon* as possible. Contrary to the previous section where a constant gain is used throughout the entire optimization process by disregarding whether we have observed a jump, this section proposes a method to control the non-diminishing step-size based upon the observable information $\mathcal{F}_k$ defined in (2.3), aiming to achieve better performance within each regime and *faster* adaptation between different regimes.

5.2.1 Model Assumptions

*Assumption C.*5 (Abstraction of a regime that lasts a long duration of time). Assume the following:

(i) $\theta_k^* = \theta^*$ for all k.

(ii) $\boldsymbol{g}_k(\theta) = \boldsymbol{H}(\theta - \theta^*)$ for some symmetric and positive-definite matrix $\boldsymbol{H}$ for all k.

(iii) ξ_k are i.i.d. with mean $\mathbf{0}$ and bounded covariance $\boldsymbol{V}$ for all k.

C.6 is more general than C.5.

*Assumption C.*6 (Abstraction of regime-switch). Assume the following:

(i) $\boldsymbol{g}_k(\theta) = \boldsymbol{H}(\theta - \theta_k^*)$ for some symmetric and positive-definite matrix $\boldsymbol{H}$ fo all k.

(ii) ξ_k are i.i.d. with mean $\mathbf{0}$ and bounded covariance $\boldsymbol{V}$ for all k.

(iii) Suppose C.3 holds.

Let us first provide some discussions on the Assumption C.5.

- At first glance, C.5 reduces the time-varying problem (3.1) to the classical SA problem of minimizing a fixed loss function reviewed in Section 2.1. Nonetheless, this stationarity assumption is imposed to facilitate the discussion of the base case in Subsection 5.2.2. Later on, the exposition in Subsection 5.2.2 will be readily extended to a more general case in Subsection 5.2.3.

- The Markovian process $\{\hat{\theta}_k\}$ generated from (4.32) does *not* approach θ^* beyond a certain distance, when the gain is held constant. C.5 is imposed so that we can devise a strategy to decide when to decrease the gain to further approach θ^*.

- Another reason to impose C.5, for the time being, is as follows. Gain selection is not a problem exclusive to nonstationarity tracking. For the stationary/fixed setting in classic SA literature, the decaying sequence $a_k = O(1/k)$ may not be desirable for practical usage, even though it is proven to be asymptotically optimal (in minimizing the trace of the limiting covariance of $\hat{\theta}_k$). Worse still, even when we pick the decaying gain sequence with the $O(1/k)$ decaying rate, the constant sitting in front of $1/k$ still drastically affects our estimation if it is misspecified and is very sensitive to the initialization. In fact, (1) the absolute value of the gain plays a more important role than the convergence rate to zero, especially when we only have limited resources to run a *finite* number of iterations, and (2) the $O(1/k)$ sequence decays *extremely slowly* to zero for a large k. Based on the two observations, we take the gain to be constant, yet small, to mimic the behavior of the estimates $\hat{\theta}_k$ within *finite* iterations. Moreover, the iterates reaching the vicinity of θ^* quickly within finite iterations is much more important than convergence after potentially infinite iterations.

Let us also mention a few subtitles implied from Assumption C.6.

- The Markovian process $\{\hat{\theta}_k\}$ becomes more difficult to analyze due to the randomness of start_s and end_s. To keep it concise, we still assume that $(\text{end}_\mathsf{s} - \text{start}_\mathsf{s})$ is sufficiently long w.p.1.; hence, the following assumption.

- C.6 is very similar to those imposed in Subsection 5.1.2, except that we do not allow dependence of $\boldsymbol{H}$ and $\boldsymbol{V}$ on the regime s here. This is because, both $\boldsymbol{H}$ and $\boldsymbol{V}$ being constant, plays an important role in Algorithm 3 to appear.

Overall, the discussion here applies to the following algorithm:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_{(\mathsf{s})}\hat{\boldsymbol{g}}_k^{\mathrm{SG}}(\hat{\theta}_k), \quad \text{start}_\mathsf{s} \leq k \leq \text{end}_\mathsf{s} - 1, \tag{5.15}$$

as long as $(\text{end}_\mathsf{s} - \text{start}_\mathsf{s})$ in C.3 is sufficiently large. (5.15) is a straightforward extension of (4.32), and let us call (5.15) "SGD with regime-wise-constant gain." For brevity's sake, we suppress the dependence of $\hat{\theta}_k$ generated by (4.32) on a under C.5, or the dependence of $\hat{\theta}_k$ generated by (5.15) on $a_{(\mathsf{s})}$ under C.6. Consequently, $\hat{\theta}_k$ exhibits a relatively short (compared to the entire regime duration) transient phase and a relatively long steady-state phase, as noted in Subsection 5.1.2.

Moreover, part (i) in both C.5 and C.6 can be relaxed to [83, Assumption B(ii)], i.e., $\boldsymbol{g}_k(\theta) = \boldsymbol{H}(\theta - \theta_k^*) + O(\|(\theta - \theta_k^*)\|^2)$. Also, part (ii) in both C.5 and C.6 can be relaxed to [83, Assumption A(iii)]; i.e., $\|\xi_k(\hat{\theta}_k)\|^2$ can be upper bounded by a linear function of $\|\hat{\theta}_k - \theta_k^*\|$ w.p.1. We use stronger assumptions to present the results more elegantly.

5.2.2 Base Case: Detection of Transient Phase and Steady-State Phase

Even though the constant-gain SA iterates will approach θ^* in neither a.s. nor m.s. sense, practitioners still implement SA with a constant gain [28, 106]. As mentioned in Section 2.2, during the transient phase, the constant-gain SA estimate generated from (4.32) promptly moves towards the desired region and forgets the initial condition exponentially fast. Then during the steady-state phase, the estimate oscillates around θ^* at a region of radius $O(\sqrt{a})$. The trade-off is obvious that a larger value of a shortens the transient phase, yet simultaneously enlarges the radius of the steady-state phase. Understanding the transition between the transient phase and the steady-state phase enables us to enhance the empirical performance of the constant-gain algorithm.

The key puzzle in designing adaptive gain is to determine a statistical test to check the stationarity of the iterates generated from (4.32). The motivation for the stationarity check comes from a gain-tuning rule in *deterministic* optimization: increase/decrease the gain if $[\boldsymbol{g}(\hat{\theta}_{k+1})]^T[\boldsymbol{g}(\hat{\theta}_k)]$ is positive/negative. In a deterministic scenario with $\xi_k = 0$ for all k, the recursion $\hat{\theta}_{k+1} = \hat{\theta}_k - a\boldsymbol{g}(\hat{\theta}_k) = \hat{\theta}_k - a\boldsymbol{H}(\hat{\theta}_k - \theta^*)$ converges to θ^* as long as the gain sequence a is smaller than $\lambda_{\min}(\boldsymbol{H})$ after some k. Note that the convergence of $\hat{\theta}_k$ to θ^* under *noise-free* scenario does *not* require the constant gain a to go to zero.

It seems natural to extend the above to use $[\hat{\boldsymbol{g}}_{k+1}(\hat{\theta}_{k+1})]^T[\hat{\boldsymbol{g}}_k(\hat{\theta}_k)]$ as an indicator for both the transient phase and the steady-state phase. However, we have to handle the *noise* ξ_k in SA problem setting. During the transient phase, the observations $\{\hat{\boldsymbol{g}}_k(\hat{\theta}_k)\}$ are auto-correlated as successive gradient observations that are *roughly* pointing to the same

direction. During the steady-state phase, successive gradient estimates *tend* to point to opposite directions. To shorten the transient phase, we are better off increasing the gain a by a factor of η_+. To move towards the optimum during the steady-state phase, it is advisable to decrease the gain by a factor of η_-. To compensate for the noise effect, we will alternatively use the running average of the inner product of the successive gradient across a sliding window. References [53, Sect. 2] and [84] provide a high-level discussion on this statistic. Nevertheless, little work has been done in determining the critical values to draw a confident conclusion of either a transient or steady-state phase.

Theorem 5.2.1 (Detection of Transient Phase and Steady-State Phase)**.** *Under C.5, let us pick the gain a such that*

$$a < \mathscr{L}^{-1}, \tag{5.16}$$

where $\lambda_{\max}(\boldsymbol{H}) = \mathscr{L}$.

(1) During the steady-state *phase for* large k*, we have*

$$\mathbb{E}\left\{[\hat{\boldsymbol{g}}_k(\hat{\theta}_k)]^T \hat{\boldsymbol{g}}_{k-1}(\hat{\theta}_{k-1})\right\} \approx -a\mathrm{tr}(\boldsymbol{H}\boldsymbol{V}) + O(a^2), \text{ for large } k \text{ and for (5.16)}, \tag{5.17}$$

and

$$\mathbb{V}\left\{\frac{1}{\mathsf{w}}\sum_{k-\mathsf{w}+1}^{k}[\hat{\boldsymbol{g}}_i(\hat{\theta}_i)]^T[\hat{\boldsymbol{g}}_{i-1}(\hat{\theta}_{i-1})]\right\} \leq \frac{1}{\mathsf{w}}O(a), \text{ for large } k \text{ and for (5.16)}, \tag{5.18}$$

where $\mathsf{w} > 0$ is an arbitrary window size.

(2) During the transient *phase for* small $k \geq 1$*, we have*

$$\mathbb{E}\left\{[\hat{\boldsymbol{g}}_k(\hat{\theta}_k)]^T \hat{\boldsymbol{g}}_{k-1}(\hat{\theta}_{k-1})\right\} \leq \hat{\theta}_0^T \boldsymbol{H}^2 \hat{\theta}_0 - a\left[\hat{\theta}_0^T \boldsymbol{H}^3 \hat{\theta}_0 + \mathrm{tr}(\boldsymbol{H}\boldsymbol{V})\right] + O(a^2),$$

for a satisfying (5.16). (5.19)

Proof of Theorem 5.2.1. Assume that $\theta^* = \mathbf{0}$ w.l.o.g., as the following discussion remains to be valid if $\hat{\theta}_k$ is replaced by $(\hat{\theta}_k - \theta^*)$ for a nonzero θ^*.

Under C.5, we can rewrite (4.32) as follows:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a\boldsymbol{H}\hat{\theta}_k - a\boldsymbol{\xi}_k = (\boldsymbol{I} - a\boldsymbol{H})\hat{\theta}_k - a\boldsymbol{\xi}_k, \quad \text{for } k \geq 0, \tag{5.20}$$

$$\implies \quad \hat{\theta}_k = (\boldsymbol{I} - a\boldsymbol{H})^k \hat{\theta}_0 - a\sum_{i=0}^{k-1}\left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-1-i}\right] \quad \text{for } k \geq 1. \tag{5.21}$$

From (5.21) we know that the Markovian process $\hat{\theta}_k$ generated from (4.32) is comprised of a deterministic part $(\boldsymbol{I} - a\boldsymbol{H})^k\hat{\theta}_0$ (assuming that there is no randomness in $\hat{\theta}_0$) and a stochastic part $-a\sum_{i=0}^{k-1}[(\boldsymbol{I} - a\boldsymbol{H})^i\boldsymbol{\xi}_{k-1-i}]$, which has a mean of $\mathbf{0}$ under C.5.

With a gain satisfying (5.16), the deterministic part goes to $\mathbf{0}$ exponentially as k grows, and the stochastic part converges in law to the stationary process $-a\sum_{i=0}^{\infty}(\boldsymbol{I} - a\boldsymbol{H})^i\boldsymbol{\xi}_{k-1-i}$. Resultingly, during the transient phase for *small* k, the linear convergence of the deterministic part is dominating compared with the stochastic part with a mean of zero; then during the steady-state phase for *large* k, the oscillating characteristic of the

stationary process $-a\sum_{i=0}^{\infty}(\boldsymbol{I}-a\boldsymbol{H})^i\boldsymbol{\xi}_{k-1-i}$ dominates compared to the deterministic part that decays to $\mathbf{0}$ exponentially fast in k.

Similarly, we can also rewrite the *noisy* gradient observation as follows:

$$\begin{aligned} & \hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) = \boldsymbol{H}\hat{\boldsymbol{\theta}}_k + \boldsymbol{\xi}_k \\ \Longrightarrow \quad & \hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) = \boldsymbol{H}(\boldsymbol{I}-a\boldsymbol{H})^k\hat{\boldsymbol{\theta}}_0 + \boldsymbol{\xi}_k - a\boldsymbol{H}\sum_{i=0}^{k-1}\left[(\boldsymbol{I}-a\boldsymbol{H})^i\boldsymbol{\xi}_{k-1-i}\right], \quad (5.22) \end{aligned}$$

where the implication in (5.22) uses (5.21) directly. From (5.22), we see that $\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ is comprised of a deterministic part $\boldsymbol{H}(\boldsymbol{I}-a\boldsymbol{H})^k\hat{\boldsymbol{\theta}}_0$ (assuming that there is no randomness in $\hat{\boldsymbol{\theta}}_0$) and a stochastic part $\{\boldsymbol{\xi}_k - a\boldsymbol{H}\sum_{i=0}^{k-1}[(\boldsymbol{I}-a\boldsymbol{H})^i\boldsymbol{\xi}_{k-1-i}]\}$, which has a mean of $\mathbf{0}$ under C.5. Again, with a constant gain a such that (5.16) holds, we see the deterministic part goes to $\mathbf{0}$ exponentially as k grows, and the stochastic part converges in law to a stationary process $\boldsymbol{\xi}_k - a\boldsymbol{H}\sum_{i=0}^{\infty}(\boldsymbol{I}-a\boldsymbol{H})^i\boldsymbol{\xi}_{k-1-i}$.

Let us consider the *steady-state* phase for *large* k. The multivariate moving-average process $-a\sum_{i=0}^{\infty}(\boldsymbol{I}-a\boldsymbol{H})^i\boldsymbol{\xi}_{k-1-i}$ is mean zero. Denote the covariance matrix for $-a\sum_{i=0}^{\infty}(\boldsymbol{I}-a\boldsymbol{H})^i\boldsymbol{\xi}_{k-1-i}$ as $\widetilde{\boldsymbol{V}}$. For large k, $\widetilde{\boldsymbol{V}}$ satisfies the following:

$$\widetilde{\boldsymbol{V}} = (\boldsymbol{I}-a\boldsymbol{H})\widetilde{\boldsymbol{V}}(\boldsymbol{I}-a\boldsymbol{H}) + a^2\boldsymbol{V}, \text{ for } a \text{ satisfying (5.16)}, \quad (5.23)$$

by taking the variance on both sides of (5.20) and then letting $k \to \infty$. The solution to (5.23) can be explicitly expressed as:

$$\widetilde{\boldsymbol{V}} = a^2 \sum_{i=0}^{\infty} (\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{V} (\boldsymbol{I} - a\boldsymbol{H})^i. \tag{5.24}$$

Meanwhile, the multivariate moving-average process $\boldsymbol{\xi}_k - a\boldsymbol{H} \sum_{i=0}^{\infty} (\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-1-i}$ also has a mean of zero. Denote the covariance matrix for $\boldsymbol{\xi}_k - a\boldsymbol{H} \sum_{i=0}^{\infty} (\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-1-i}$ as $\overline{\boldsymbol{V}}_k$. For large k, $\overline{\boldsymbol{V}}_k$ satisfies the following:

$$\overline{\boldsymbol{V}}_k = a^2 \boldsymbol{H} \left\{ \sum_{i=0}^{\infty} \left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{V} (\boldsymbol{I} - a\boldsymbol{H})^i \right] \right\} \boldsymbol{H} + \boldsymbol{V} = \boldsymbol{H} \widetilde{\boldsymbol{V}} \boldsymbol{H} + \boldsymbol{V}, \text{ for (5.16)}, \tag{5.25}$$

by taking the variance on both sides of (5.22) and using (5.24). Moreover, for $k \geq l$, the covariance of $\hat{\boldsymbol{g}}_k(\hat{\theta}_k)$ and $\hat{\boldsymbol{g}}_l(\hat{\theta}_l)$ for large l is

$$\overline{\boldsymbol{V}}_{k:l} \equiv \mathbb{C}(\hat{\boldsymbol{g}}_k(\hat{\theta}_k), \hat{\boldsymbol{g}}_l(\hat{\theta}_l)) = (\boldsymbol{I} - a\boldsymbol{H})^{k-l} \overline{\boldsymbol{V}}_k. \tag{5.26}$$

When (5.16) holds, we have $\|\boldsymbol{I} - a\boldsymbol{H}\| = 1 - a\mathscr{C} \in (0, 1)$, where $\mathscr{C} = \lambda_{\min}(\boldsymbol{H})$. Hence,

$$\|\overline{\boldsymbol{V}}_{k:l}\| \leq (1 - a\mathscr{C})^{k-l} \|\overline{\boldsymbol{V}}_k\| \tag{5.27}$$

where the number $(1-a\mathscr{C})^{k-l}$ arises due to $\|(\boldsymbol{I}-a\boldsymbol{H})^{k-l}\| \leq \|\boldsymbol{I}-a\boldsymbol{H}\|^{k-l} = (1-a\mathscr{C})^{k-l}$.

Using (5.22), we have the following approximation:

$$
\begin{aligned}
& [\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)]^T \hat{\boldsymbol{g}}_{k-1}(\hat{\boldsymbol{\theta}}_{k-1}) \\
\approx\ & \left\{ \boldsymbol{\xi}_k - a\boldsymbol{H} \sum_{i=0}^{k-1} \left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-1-i} \right] \right\}^T \left\{ \boldsymbol{\xi}_{k-1} - a\boldsymbol{H} \sum_{i=0}^{k-2} \left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-2-i} \right] \right\} \\
=\ & \left\{ \boldsymbol{\xi}_k - a\boldsymbol{H}\boldsymbol{\xi}_{k-1} - a\boldsymbol{H} \sum_{i=1}^{k-1} \left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-1-i} \right] \right\}^T \\
& \quad \cdot \left\{ \boldsymbol{\xi}_{k-1} - a\boldsymbol{H} \sum_{i=0}^{k-2} \left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-2-i} \right] \right\} \\
=\ & \left\{ \boldsymbol{\xi}_k - a\boldsymbol{H}\boldsymbol{\xi}_{k-1} - a\boldsymbol{H}(\boldsymbol{I} - a\boldsymbol{H}) \sum_{i=0}^{k-2} \left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-2-i} \right] \right\}^T \\
& \quad \cdot \left\{ \boldsymbol{\xi}_{k-1} - a\boldsymbol{H} \sum_{i=0}^{k-2} \left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-2-i} \right] \right\}, \quad \text{for large } k \text{ and } a \text{ satisfying (5.16),}
\end{aligned}
\tag{5.28}
$$

where the first approximation is claimed after discarding the deterministic part in (5.22) for *large* k. Combining the above observations, we have the following:

$$
\begin{aligned}
& \mathbb{E}\left\{ [\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)]^T \hat{\boldsymbol{g}}_{k-1}(\hat{\boldsymbol{\theta}}_{k-1}) \right\} \\
& = \mathbb{E}\left(-a\boldsymbol{\xi}_{k-1}^T \boldsymbol{H} \boldsymbol{\xi}_{k-1} \right) \\
& \quad + \mathbb{E}\left(\left\{ -a \sum_{i=0}^{k-2} \left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-2-i} \right] \right\}^T (\boldsymbol{I} - a\boldsymbol{H})\boldsymbol{H}^2 \left\{ -a \sum_{i=0}^{k-2} \left[(\boldsymbol{I} - a\boldsymbol{H})^i \boldsymbol{\xi}_{k-2-i} \right] \right\} \right) \\
& \approx -a\mathrm{tr}(\boldsymbol{H}\boldsymbol{V}) + \mathrm{tr}((\boldsymbol{I} - a\boldsymbol{H})\boldsymbol{H}^2 \widetilde{\boldsymbol{V}}) \\
& = -a\mathrm{tr}(\boldsymbol{H}\boldsymbol{V}) + \mathrm{tr}(\boldsymbol{H}^2 \widetilde{\boldsymbol{V}}) - a\mathrm{tr}(\boldsymbol{H}^3 \widetilde{\boldsymbol{V}}) \\
& = -a\mathrm{tr}(\boldsymbol{H}\boldsymbol{V}) + O(a^2), \quad \text{for large } k \text{ and for } a \text{ satisfying (5.16),}
\end{aligned}
\tag{5.29}
$$

where the approximation uses (5.23), (5.22) and C.5, and the last equation is due to the coefficient a^2 on the r.h.s. of (5.24). Furthermore, we also have:

$$\mathbb{C}\left([\hat{\boldsymbol{g}}_k(\hat{\theta}_k)]^T\hat{\boldsymbol{g}}_{k-1}(\hat{\theta}_{k-1}), [\hat{\boldsymbol{g}}_l(\hat{\theta}_l)]^T\hat{\boldsymbol{g}}_{l-1}(\hat{\theta}_{l-1})\right) = O(a(1-a\mathscr{C})^{k-l}), \tag{5.30}$$

which follows from (5.27). Then (5.18) immediately follows.

Let us consider the *transient* phase for *small* k. We have the following observation:

$$\begin{aligned}
&\mathbb{E}\left\{[\hat{\boldsymbol{g}}_k(\hat{\theta}_k)]^T\hat{\boldsymbol{g}}_{k-1}(\hat{\theta}_{k-1})\right\} \\
&= \hat{\theta}_0^T(\boldsymbol{I}-a\boldsymbol{H})^k\boldsymbol{H}^2(\boldsymbol{I}-a\boldsymbol{H})^{k-1}\hat{\theta}_0 - \mathbb{E}\left(a\boldsymbol{\xi}_{k-1}^T\boldsymbol{H}\boldsymbol{\xi}_{k-1}\right) \\
&\quad + \mathbb{E}\left(\left\{-a\sum_{i=0}^{k-2}\left[(\boldsymbol{I}-a\boldsymbol{H})^i\boldsymbol{\xi}_{k-2-i}\right]\right\}^T(\boldsymbol{I}-a\boldsymbol{H})\boldsymbol{H}^2\left\{-a\sum_{i=0}^{k-2}\left[(\boldsymbol{I}-a\boldsymbol{H})^i\boldsymbol{\xi}_{k-2-i}\right]\right\}\right) \\
&= \hat{\theta}_0^T(\boldsymbol{I}-a\boldsymbol{H})^k\boldsymbol{H}^2(\boldsymbol{I}-a\boldsymbol{H})^{k-1}\hat{\theta}_0 - a\mathrm{tr}(\boldsymbol{H}\boldsymbol{V}) + a^2\sum_{i=0}^{k-2}\mathrm{tr}((\boldsymbol{I}-a\boldsymbol{H})^{2i+1}\boldsymbol{H}^2\boldsymbol{V}),
\end{aligned}$$

$$\text{for } k \geq 2, \tag{5.31}$$

where the binomial series $(\boldsymbol{I}-a\boldsymbol{H})^k = \sum_{i=0}^k \binom{k}{i}(a\boldsymbol{H})^i$ for $k \geq 0$. For $k=1$, we have

$$\begin{aligned}
&\mathbb{E}\left\{[\hat{\boldsymbol{g}}_k(\hat{\theta}_k)]^T\hat{\boldsymbol{g}}_{k-1}(\hat{\theta}_{k-1})\right\} \\
&= \hat{\theta}_0^T(\boldsymbol{I}-a\boldsymbol{H})\boldsymbol{H}^2\hat{\theta}_0 - a\mathrm{tr}(\boldsymbol{H}\boldsymbol{V}) \\
&= \hat{\theta}_0^T\boldsymbol{H}^2\hat{\theta}_0 - a\hat{\theta}_0^T\boldsymbol{H}^3\hat{\theta}_0 - a\mathrm{tr}(\boldsymbol{H}\boldsymbol{V}) \quad \text{for } a \text{ satisfying (5.16).}
\end{aligned} \tag{5.32}$$

For $k = 2$, we have

$$\begin{aligned}
&\mathbb{E}\left\{[\hat{\boldsymbol{g}}_k(\hat{\theta}_k)]^T\hat{\boldsymbol{g}}_{k-1}(\hat{\theta}_{k-1})\right\} \\
&= \hat{\theta}_0^T(\boldsymbol{I} - a\boldsymbol{H})^2\boldsymbol{H}^2(\boldsymbol{I} - a\boldsymbol{H})\hat{\theta}_0 - a\mathrm{tr}(\boldsymbol{HV}) + O(a^2) \\
&= \hat{\theta}_0^T\boldsymbol{H}^2\hat{\theta}_0 - 3a\hat{\theta}_0^T\boldsymbol{H}^3\hat{\theta}_0 - a\mathrm{tr}(\boldsymbol{HV}) + O(a^2), \qquad \text{for } a \text{ satisfying (5.16).} \qquad (5.33)
\end{aligned}$$

In general, for small $k \geq 1$, the magnitude deterministic part should dominate the magnitude of the mean-zero stochastic part, and (5.19) holds. □

Based on Theorem 5.2.1, we propose the following strategy to adapt the gain sequence. Let us recursively define a sequence of "critical" times $\{\tilde{\kappa}_s\}$ such that:

$$\begin{aligned}
\tilde{\kappa}_{s+1} = \inf\Bigg\{ k > \tilde{\kappa}_s \Bigg| &\frac{1}{k-\tilde{\kappa}_s}\sum_{i=\tilde{\kappa}_s+1}^{k}[\hat{\boldsymbol{g}}_i(\hat{\theta}_i)]^T[\hat{\boldsymbol{g}}_{i-1}(\hat{\theta}_{i-1})] \text{ is either } \leq -a\mathrm{tr}\left(\widehat{\boldsymbol{HV}}\right) \\
&\text{or } \geq \hat{\theta}_k^T\widehat{\boldsymbol{H}}^2\hat{\theta}_k - a\left[\hat{\theta}_k^T\widehat{\boldsymbol{H}}^3\hat{\theta}_k + \mathrm{tr}(\widehat{\boldsymbol{HV}})\right]\Bigg\}, \\
\text{with } \tilde{\kappa}_0 = 0, \qquad & (5.34)
\end{aligned}$$

where $\widehat{\boldsymbol{H}}$ and $\widehat{\boldsymbol{V}}$ are the estimates for $\boldsymbol{H}$ and $\boldsymbol{V}$, respectively. How to construct $\widehat{\boldsymbol{H}}$ and $\widehat{\boldsymbol{V}}$ will be discussed momentarily. Correspondingly, the gain sequence is defined by:

$$a_{[\mathrm{s}+1]} = \begin{cases} \eta_- a_{[\mathrm{s}]}, \text{ if } \frac{1}{k-\tilde{\kappa}_\mathrm{s}} \sum_{i=\tilde{\kappa}_\mathrm{s}+1}^{k} [\hat{\boldsymbol{g}}_i(\hat{\theta}_i)]^T [\hat{\boldsymbol{g}}_{i-1}(\hat{\theta}_{i-1})] \leq -a\mathrm{tr}\,(\boldsymbol{H}\boldsymbol{V}), \\ \eta_+ a_{[\mathrm{s}]}, \\ \quad \text{if } \frac{1}{k-\tilde{\kappa}_\mathrm{s}} \sum_{i=\tilde{\kappa}_\mathrm{s}+1}^{k} [\hat{\boldsymbol{g}}_i(\hat{\theta}_i)]^T [\hat{\boldsymbol{g}}_{i-1}(\hat{\theta}_{i-1})] \geq \hat{\theta}_k^T \widehat{\boldsymbol{H}}^2 \hat{\theta}_k - a\left[\hat{\theta}_k^T \widehat{\boldsymbol{H}}^3 \hat{\theta}_k + \mathrm{tr}(\widehat{\boldsymbol{H}}\widehat{\boldsymbol{V}})\right], \end{cases} \tag{5.35}$$

where $a = a_{[\mathrm{s}]}$ for $\tilde{\kappa}_\mathrm{s} \leq k < \tilde{\kappa}_{\mathrm{s}+1}$. Unfortunately, we do not have any quantification regarding the Type-I and Type-II errors for the phase detection in (5.34) at the moment.

Estimation of Hessian Information and Error Covariance

We need both $\boldsymbol{H}$ and $\boldsymbol{V}$ to perform gain adaptation (5.35), yet they are unknown. Let us briefly obtain $\widehat{\boldsymbol{H}}$ and $\widehat{\boldsymbol{V}}$ through the observable information $\mathcal{F}_{k+1}$. We borrow the SP idea in [105] to construct $\widehat{\boldsymbol{H}}$. Here we will slightly alter the recursion (4.32) into the following:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a\frac{\hat{\boldsymbol{g}}_k(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k) + \hat{\boldsymbol{g}}_k(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)}{2}, \tag{5.36}$$

where the setup for c_k and $\boldsymbol{\Delta}_k$ are the same as that in (3.4). At the cost of two measurements at each k, we can estimate $\boldsymbol{H}$ recursively as follows:

$$\begin{aligned}\widehat{\boldsymbol{H}}_k = &\frac{k}{k+1}\widehat{\boldsymbol{H}}_{k-1} + \frac{1}{4c_k(k+1)}[\hat{\boldsymbol{g}}_k(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k) - \hat{\boldsymbol{g}}_k(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)]\boldsymbol{\Delta}_k^{-T} \\ &+ \frac{1}{4c_k(k+1)}\boldsymbol{\Delta}_k^{-1}[\hat{\boldsymbol{g}}_k(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k) - \hat{\boldsymbol{g}}_k(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)]^T,\ k = 1, 2, \cdots \end{aligned} \tag{5.37}$$

where $\boldsymbol{\Delta}^{-T} = (\boldsymbol{\Delta}^{-1})^T$. For more details, see [105] or (A.6) in Appendix A. Note that the initialization for (5.37) may be a scale matrix ($\text{scale} \cdot \boldsymbol{I}_p$ for $\text{scale} > 0$), or some other positive-definite matrix reflecting available information (e.g., if one knows that θ elements will have very different magnitudes, then the initialization may be chosen to approximately scale for the differences). Similarly, we estimate $\boldsymbol{V}$ recursively as follows:

$$\begin{aligned}\hat{\boldsymbol{V}}_k = &\frac{k}{k+1}\hat{\boldsymbol{V}}_{k-1} \\ &+ \frac{1}{(k+1)}\left[\hat{\boldsymbol{g}}_k(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k) - \hat{\boldsymbol{g}}_k(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)\right]\left[\hat{\boldsymbol{g}}_k(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k) - \hat{\boldsymbol{g}}_k(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)\right]^T. \end{aligned} \tag{5.38}$$

5.2.2.1 Summary of Adapted Gain-Tuning Algorithm

Let us summarize the aforementioned procedure, including gain adaptation and the estimation of $\boldsymbol{H}$ and $\boldsymbol{V}$ in Algorithm 3 below.

Algorithm 3 Adaptive Gain Selection for Change Detection Using *Two*-Measurements $\hat{\boldsymbol{g}}_k(\hat{\theta}_k \pm c_k\boldsymbol{\Delta}_k)$ at a Time

Input: initial gain magnitude a, $\hat{\theta}_0$, increase ratio η_+, decrease ratio η_-.

set $\mathrm{s} = 0$ and $\tilde{\kappa}_0 = 0$.
for $k \geq 1$ or $k \in \{1, \cdots, K\}$ **do** ▷ K is the horizon over which we need to perform tracking.
  collect $\hat{\boldsymbol{g}}_k(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k)$ and $\hat{\boldsymbol{g}}_k(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)$. ▷ We may let c_k be the desired minimal change in components of $\hat{\theta}_k$, and generate $\boldsymbol{\Delta}_k$ from symmetric Bernoulli ± 1 distribution.
  update $\widehat{\boldsymbol{H}}_k$ and $\widehat{\boldsymbol{V}}_k$ using (5.37) and (5.38) respectively.
  if $\frac{1}{k-\tilde{\kappa}_s+1}\sum_{i=\tilde{\kappa}_s+1}^{k}\{[\hat{\boldsymbol{g}}_i(\hat{\theta}_i)]^T[\hat{\boldsymbol{g}}_{i-1}(\hat{\theta}_{i-1})]\} < -a\mathrm{tr}(\widehat{\boldsymbol{H}}_k\widehat{\boldsymbol{V}}_k)$ **then**
    decrease gain a by a factor of η_-.
    set $\mathrm{s} \leftarrow \mathrm{s} + 1$.
  else if $\frac{1}{k-\tilde{\kappa}_s+1}\sum_{i=\tilde{\kappa}_s+1}^{k}\{[\hat{\boldsymbol{g}}_i(\hat{\theta}_i)]^T[\hat{\boldsymbol{g}}_{i-1}(\hat{\theta}_{i-1})]\} \geq \hat{\theta}_k^T\widehat{\boldsymbol{H}}_k^2\hat{\theta}_k - a\left[\hat{\theta}_k^T\widehat{\boldsymbol{H}}_k^3\hat{\theta}_k + \mathrm{tr}(\widehat{\boldsymbol{H}}_k\widehat{\boldsymbol{V}}_k)\right]$ **then**
    increase gain a by a factor of η_+.
    set $\mathrm{s} \leftarrow \mathrm{s} + 1$.
  end if
  update $\hat{\theta}_k$ using (5.36).
Output: $\hat{\theta}_k$.
end for

An Example for Adaptive Gain

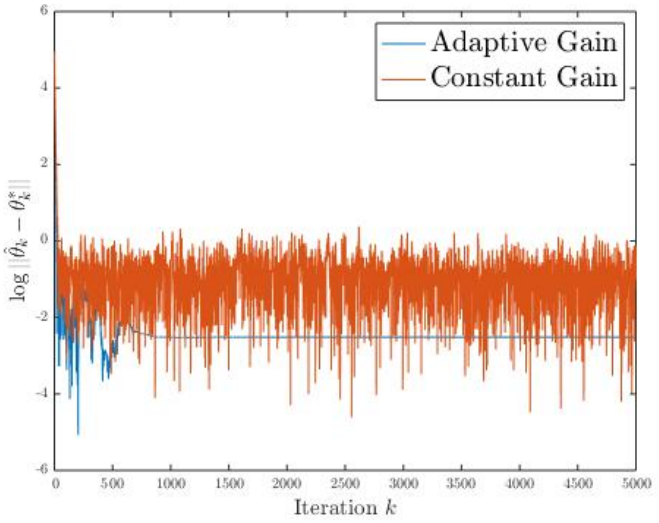
Here, we again consider $p = 2$. The loss function is $f(\theta) = (\theta - \theta^*)^T \boldsymbol{H}(\theta - \theta^*)/2$, and the gradient function is $\boldsymbol{g}(\theta) = \boldsymbol{H}(\theta - \theta^*)$. Again, $\boldsymbol{H}$ is constructed as $\boldsymbol{H} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^T$ in (3.42), where $\boldsymbol{P}$ is (randomly generated) orthogonal, and $\boldsymbol{D}$ is diagonal with diagonal entries 30 and 5. For simplicity, we select $\theta^* = \mathbf{0}$. We pick an increase ratio of $\eta_+ = 1.1$ and a decrease ratio $\eta_- = 0.9$. The observational noise again follows i.i.d. $\mathrm{Normal}(\mathbf{0}, \sigma_1^2 \boldsymbol{I}_p)$ with $\sigma_1 = 10$. We use an initialization $(100 \;\; 100)^T$, which is far away from $\theta^* = \mathbf{0}$. We can make the following observations from Figures 5.3 to 5.5.

- For an appropriately tuned gain, the estimates generated from the constant-gain recursion (4.32) are capable of getting close to the target and perform as well as our adaptive gain algorithm 3 (see Figure 5.3a).

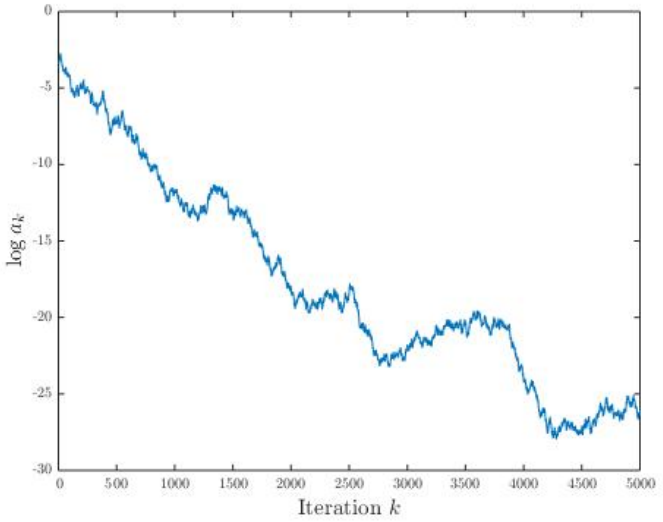
- For a gain that is too small, it takes an extremely long time for the estimates generated from constant gain recursion (4.32) to get close to the desired optimum compared with our adaptive gain algorithm 3 (see Figure 5.4a).

- For a gain that is too large, the constant-gain recursion (4.32) will migrate further and further away from the target (see Figure 5.5a).

In reality, $\mathscr{L}$ may *not* be available to the agent(s), and, the gain used in constant-gain recursion (4.32) is often *misspecified*. This further manifests the value of the data-dependent gain-tuning strategy summarized in Algorithm 3 and [125].

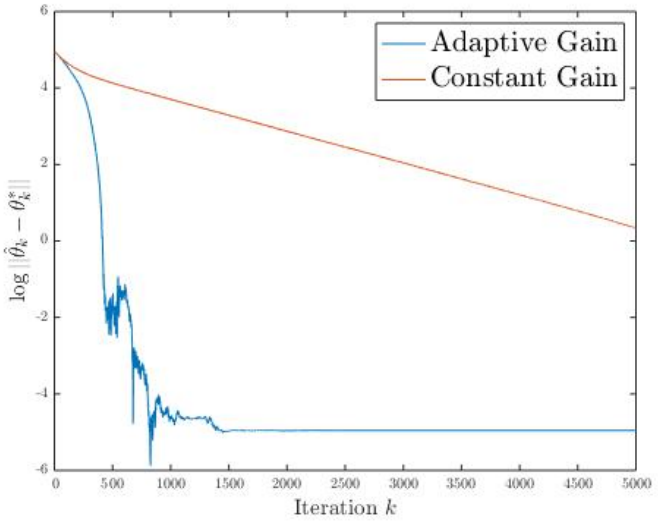


(a) Log-Euclidean-Distance Between Estimate and True Parameter

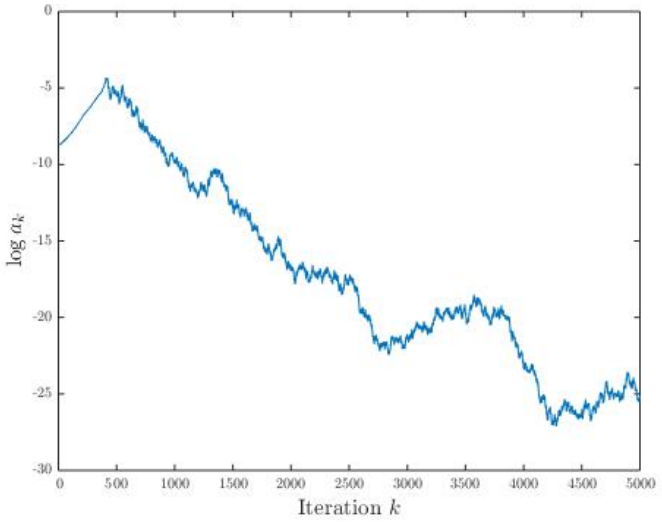


(b) Log-Magnitude of Data-Dependent Gain Sequence

Figure 5.3: A comparison of adaptive gain used in Algorithm 3 versus constant gain (4.32), both of which have gain initialized at $1/\mathscr{L} = 0.0333$.

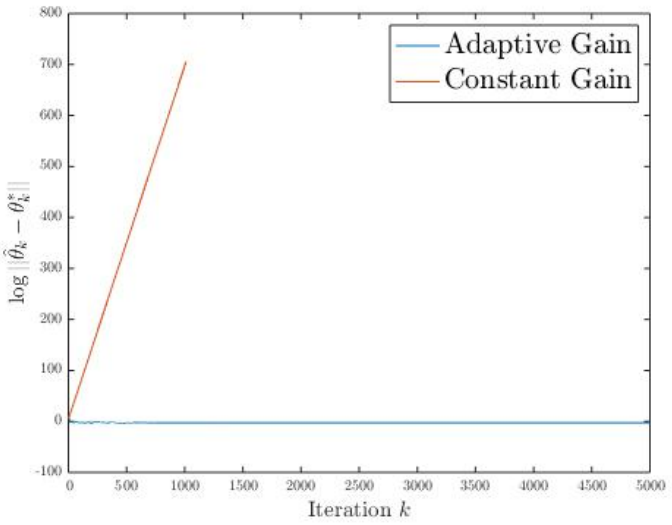


(a) Log-Euclidean-Distance Between Estimate and True Parameter

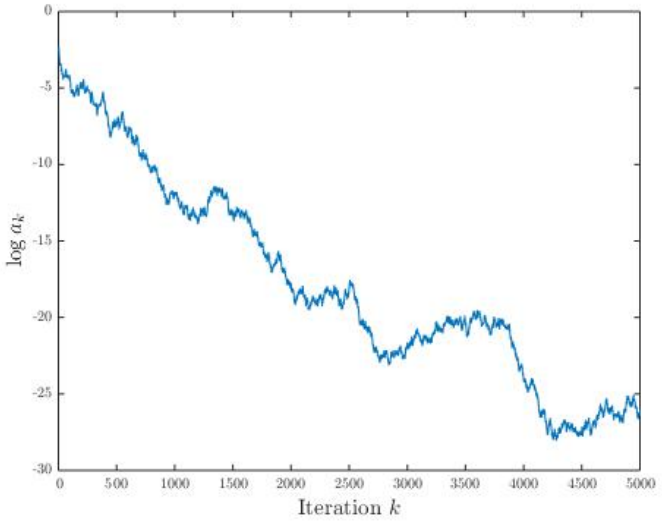


(b) Log-Magnitude of Data-Dependent Gain Sequence

Figure 5.4: A comparison of the adaptive gain used in Algorithm 3 versus the constant gain (4.32), both of which have gain initialized at $0.005/\mathscr{L} = 1.67 \times 10^{-4}$.



(a) Log-Euclidean-Distance Between Estimate and True Parameter



(b) Log-Magnitude of Data-Dependent Gain Sequence

Figure 5.5: Comparison of the adaptive gain used in Algorithm 3 versus the constant gain (4.32), both of which have gain initialized at $3/\mathscr{L} = 0.1$.

5.2.3 Building Block: Regime Change Detection With Constant Hessian

We may apply Algorithm 3 to the scenario where jump structure is allowed. This is a relatively short section as it directly applies Algorithm 3 in Section 5.2 to a more general setting C.6 based on the following observations.

1. When regime switches from $\theta^*_{(s)}$ to $\theta^*_{(s+1)}$, there will be a phase of $\hat{\theta}_k$ steadily approaching the new estimate $\theta^*_{(s+1)}$. If an abrupt change is detected, we need to increase the gain by a factor of η_+, to achieve prompt tracking.

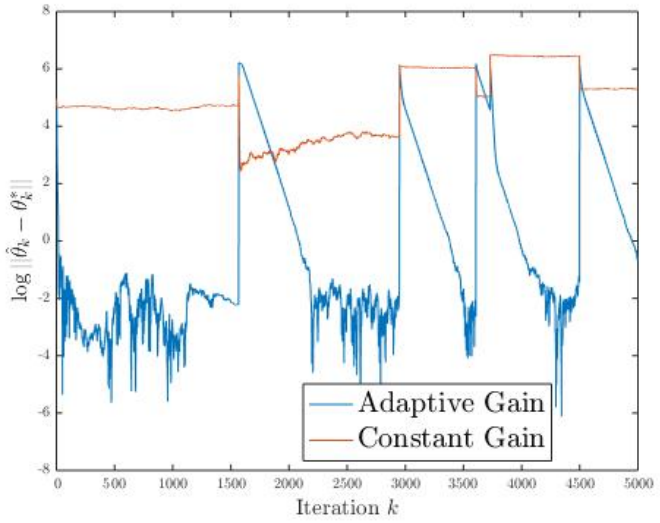
2. When some oscillating behavior is detected from the path, we need to decrease the gain by a factor of η_-, to further approach our desired target.

3. For other scenarios (no strong evidence to support a steady-state phase or the transient phase), we simply keep the gain at the most recent level. That is, the gain a is kept fixed until we gather strong evidence in favor of decreasing or increasing the gain.

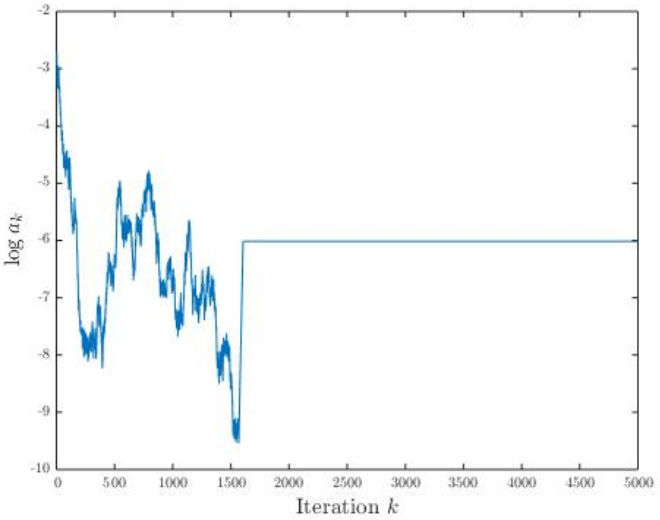
An Example of Adaptive Gain

Again consider the same numerical setup as in Subsubsection 3.3.2, except that the evolution of $\{\theta^*_k\}$ now changes to (5.13). In our simulation, the jump times for the $\{\theta^*_k\}$ are $1567, 2949, 3607, 3729$, and 4498.

We can make the following observations from Figures 5.6 to 5.7.

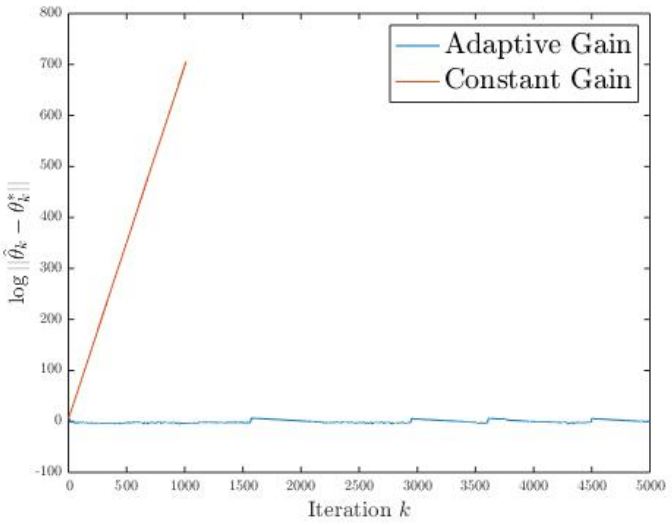


(a) Log-Euclidean-Distance Between Estimate and True Parameter

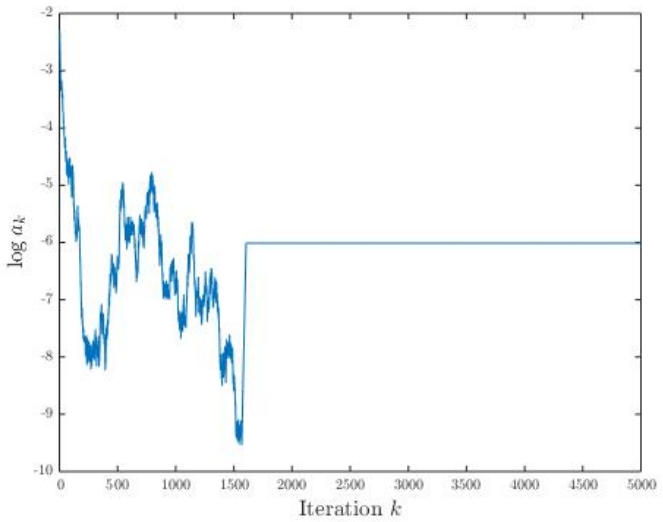


(b) Log-Magnitude of Data-Dependent Gain Sequence

Figure 5.6: A comparison of the adaptive gain used in Algorithm 3 versus constant gain (4.32), both of which have gain initialized at $2/\mathscr{L} = 0.0667$. The evolution of $\{\boldsymbol{\theta}_k^*\}$ follows (5.13).



(a) Log-Euclidean-Distance Between Estimate and True Parameter



(b) Log-Magnitude of Data-Dependent Gain Sequence

Figure 5.7: A comparison of the adaptive gain used in Algorithm 3 versus constant gain (4.32), both of which have gain initialized at $3/\mathscr{L} = 0.1$. The evolution of $\{\theta_k^*\}$ follows (5.13).

- Again, we tune the gain for the constant-gain recursion (4.32) very carefully, but this "lazy" strategy is not so robust concerning the jumps in the true $\{\theta_k^*\}$ sequence, compared to the adaptive gain algorithm listed in Algorithm 3. See Figure 5.6a. Also, from Figure 5.6b, we see that the adaptive gain tuning Algorithm 3 does increase the stepsize every time a jump arises and decreases the stepsize every time the estimate is close to the target.

- For a gain that is too large, the constant-gain recursion will migrate further and further away from the target. See Figure 5.7a. Nonetheless, the data-dependent gain-tuning enables the estimates $\hat{\theta}_k$ to stay close with the moving target.

Let us reiterate that $\mathscr{L}$ may *not* be available to the agent(s), and this further indicates the value of the data-dependent gain-tuning strategy summarized in Algorithm 3.

5.3 Concluding Remarks

Recall that Chapter 3 provides a computable error bound for non-diminishing gain SA algorithms applied in online learning and dynamic control systems, and naturally gives rise to a gain selection guidance in Algorithm 1, that depends on the strong convexity parameter $\mathscr{C}_k$, the Lipschitz continuity parameter $\mathscr{L}_k$, the noise level $\mathscr{M}_k$, and the drift level $\mathscr{B}_k$. Nonetheless, Chapter 3 only captures the average performance over possible sample paths. The practical needs to perform well in every sample path drive us to consider data-dependent gain selection strategy, which requires detecting the jump component in the

hybrid system in Section 5.1, and estimating the Hessian information and the noise level to adapt the non-diminishing gain sequence intelligently in Section 5.2. The additional restriction C.3 is imposed on the drift, mainly because the error bound discussed in Subsection 3.3.3 requires the availability of $\mathscr{C}_k$, $\mathscr{L}_k$, and $\mathscr{M}_k$, to which we may not have access in real-world applications.

We establish a framework for practical use: specifically, we can adapt our gain sequence based on our estimate of the Hessian information and the noise level. This is the key point on which our work differs from all the prior work that require unavailable information, although we impose stringent assumptions C.3 and so on. The gain adaptation algorithm is developed mainly from the observation that constant-gain SA provides a "fast transient" to the vicinity of the solution θ^*. Theorem 5.2.1 and Algorithm 3 are developed to determine the critical values to draw a confident conclusion of either a transient or steady-state phase, as little work has been done in this direction previously.

Chapter 6

A Zero-Communication Multi-Agent Problem

This chapter is an illustration of the tracking capability of SA algorithms with non-decaying gains as applied to the multi-agent multi-target surveillance mission. This problem of interest is to configure an ensemble of agents with mobile sensors over a particular region to best[1] maintain awareness of a group of targets within a specific surveillance region. This tracking problem is *dynamic* due to the motion of both the targets and the agents, and is *stochastic* due to that only inexact sensor measurements can be gathered. Given the two features, this surveillance problem fits the time-varying SA setup (3.1) perfectly, and the loss function $f_k(\cdot)$ in this chapter will be constructed in a way such that the assumptions A.1–A.4 are met. Again, there is no optimal steady-state solution due

[1]The quantification of good or bad is according to a set of mission-related metrics, such as the fraction of targets found, the accuracy of target position estimates and so on.

to the time-varying characteristic of $\{\theta_k^*\}$. In fact, this tracking problem is what motivates us to solve (3.1) using general SA algorithms (2.1) with non-decaying gain while making only modest assumptions on the error term in $\hat{\boldsymbol{g}}_k(\cdot)$ as in A.1, the underlying loss function A.2 and A.3, and the moving target as in A.4, consistent with the main focus of the entire thesis.

To ease the upcoming illustration with graphs, the discussion here will be on a two-dimensional E-N plane with "E" and "N" representing the east and north directions respectively, i.e., only the latitude and the longitude are considered. The east and the north directions can be relative to the origin of the existing geographic coordinate system, which is currently located in the Gulf of Guinea, or can be relative to any hypothetical origin of the two-dimensional plane. Nonetheless, they can be readily extended to the three-dimensional space to include the elevation (such as the altitude of the UAV or the depth of the UUV) and other higher-dimensional problems.

6.1 Base Case: One Agent and One Target

This section presents the simplest scenario where there are only one agent and one target. The notation for this base case can be readily extended to the upcoming general case with multiple targets and multiple agents.

6.1.1 Basic Tracking Setup

We first define the necessary notions for the tracking problem. Denote the state vector of the target at time τ_k as $\boldsymbol{x}_k = (x_k^{\mathrm{E}},\ x_k^{\mathrm{N}},\ \dot{x}_k^{\mathrm{E}},\ \dot{x}_k^{\mathrm{N}})^T \in \mathbb{R}^4$, where $(x_k^{\mathrm{E}},\ x_k^{\mathrm{N}})^T \in \mathbb{R}^2$ is the coordinate of the target's position at index k, and $\dot{x}_k^{\mathrm{E}}$ and $\dot{x}_k^{\mathrm{N}}$ are the magnitudes of the target's velocity in the directions of the east and the north. Similarly, the state of the agent at time τ_k will be denoted as $\boldsymbol{y}_k = (y_k^{\mathrm{E}},\ y_k^{\mathrm{N}},\ \dot{y}_k^{\mathrm{E}},\ \dot{y}_k^{\mathrm{N}})^T \in \mathbb{R}^4$.

Besides, let $\mathsf{v}_{\boldsymbol{x}}^{\max}$ and $\mathsf{v}_{\boldsymbol{y}}^{\max}$ denote the speed limits of the target and the agent respectively. They set constraints on the Euclidean norm of $(\dot{x}_k,\ \dot{x}_k)^T$ and $(\dot{y}_k,\ \dot{y}_k)^T$ respectively. Take UUVs as an example: the maximum speed is typically around 15 meters per second. We will correspondingly use one second as the unit for the sampling time τ_k. For simplicity, we will omit the unit "meters per second" for the speed limit, the unit "meter" for the distance, and the unit "seconds" for time throughout this chapter.

The available information that can be collected through the agent's sensor at time τ_k typically include the *noisy* measurement of the azimuth angle from the agent to the target defined as

$$\varphi(\boldsymbol{x},\boldsymbol{y})|_{(\boldsymbol{x},\boldsymbol{y})=(\boldsymbol{x}_k,\boldsymbol{y}_k)} = \begin{cases} \arctan\left(\dfrac{x_k^{\mathrm{N}}-y_k^{\mathrm{N}}}{x_k^{\mathrm{E}}-y_k^{\mathrm{E}}}\right), & \text{when } x_k^{\mathrm{N}} > y_k^{\mathrm{N}} \text{ and } x_k^{\mathrm{E}} > y_k^{\mathrm{E}}, \\ \arctan\left(\dfrac{x_k^{\mathrm{N}}-y_k^{\mathrm{N}}}{x_k^{\mathrm{E}}-y_k^{\mathrm{E}}}\right) + \pi, & \text{when } x_k^{\mathrm{E}} < y_k^{\mathrm{E}}, \\ \arctan\left(\dfrac{x_k^{\mathrm{N}}-y_k^{\mathrm{N}}}{x_k^{\mathrm{E}}-y_k^{\mathrm{E}}}\right) + 2\pi, & \text{when } x_k^{\mathrm{N}} < y_k^{\mathrm{N}} \text{ and } x_k^{\mathrm{E}} > y_k^{\mathrm{E}}, \end{cases} \tag{6.1}$$

and the *noisy* measurement of the range between the agent and the target denoted as

$$\rho(\boldsymbol{x},\boldsymbol{y})|_{(\boldsymbol{x},\boldsymbol{y})=(\boldsymbol{x}_k,\boldsymbol{y}_k)} = \sqrt{\left(x_k^{\mathrm{E}}-y_k^{\mathrm{E}}\right)^2+(x_k^{\mathrm{N}}-y_k^{\mathrm{N}})^2}, \tag{6.2}$$

The adjustment $\pi\,\mathbb{I}_{\{y_k^{\mathrm{E}}-x_k^{\mathrm{E}}>0\}}$ and $2\pi\mathbb{I}_{\left\{x_k^{\mathrm{N}}<y_k^{\mathrm{N}},\text{ and }y_k^{\mathrm{E}}>y_k^{\mathrm{E}}\right\}}$ in (6.1) serves to enable $\varphi(\boldsymbol{x}_k,\boldsymbol{y}_k)$ to be the direction that the agent needs to move along in order to get closer to the target. To avoid the issues arising from differentiating $\arctan(\cdot)$ function (due to its periodicity) and differentiating $\sqrt{\cdot}$ function (due to its non-differentiability at the origin) in what follows, the agent's observable information is rearranged as:

$$\boldsymbol{z}(\boldsymbol{x},\boldsymbol{y})|_{(\boldsymbol{x},\boldsymbol{y})=(\boldsymbol{x}_k,\boldsymbol{y}_k)} = \begin{pmatrix} \rho(\boldsymbol{x}_k,\boldsymbol{y}_k)\cos\left(\varphi(\boldsymbol{x}_k,\boldsymbol{y}_k)\right) \\ \rho(\boldsymbol{x}_k,\boldsymbol{y}_k)\sin\left(\varphi(\boldsymbol{x}_k,\boldsymbol{y}_k)\right) \end{pmatrix} + \boldsymbol{v}_k \in \mathbb{R}^2, \tag{6.3}$$

where $\boldsymbol{v}_k \overset{\text{i.i.d.}}{\sim} \mathrm{Normal}(\boldsymbol{0},\boldsymbol{R}_k)$ with a covariance matrix of

$$\boldsymbol{R}_k = \mathrm{diag}(10,10). \tag{6.4}$$

The covariance matrix for the measurement noise $\boldsymbol{v}_k$ in (6.4) is proposed based on the fact that the typical GPS devices nowadays is accurate anywhere within 3 to 10 meters.

After obtaining the noisy measurement (6.3), the agent needs to pick an action $\theta_k = (\dot{y}_k^{\mathrm{E}},\ \dot{y}_k^{\mathrm{N}})^T \in \mathbb{R}^2$ to determine the magnitude and the direction of its speed at time τ_k.

With the aforementioned notation, let us briefly describe the real-time tracking by iterative updating procedure.

i) At time τ_k, the target is at state $\boldsymbol{x}_k$ according to its desired motion model (which is not revealed to the agent), and the agent is at state $\boldsymbol{y}_k$.

The agent is allowed to collect noisy measurements $\boldsymbol{z}(\boldsymbol{x}_k, \boldsymbol{y}_k)$. Then the agent predicts the next possible position of the target, denoted as $\hat{\boldsymbol{x}}_{k+1|k}$ by making use of $\boldsymbol{z}(\boldsymbol{x}_k, \boldsymbol{y}_k)$, and the details will be discussed momentarily. With an a priori prediction $\hat{\boldsymbol{x}}_{k+1|k}$ for the upcoming state $\boldsymbol{x}_{k+1}$ of the target, the agent then picks a direction $\hat{\theta}_k$ such that the resulting position of the agent at time τ_{k+1} becomes

$$\begin{pmatrix} y^{\mathrm{E}}_{k+1} \\ \\ y^{\mathrm{N}}_{k+1} \end{pmatrix} = \begin{pmatrix} y^{\mathrm{E}}_{k} \\ \\ y^{\mathrm{N}}_{k} \end{pmatrix} + (\tau_{k+1} - \tau_k)\hat{\theta}_k, \text{ for } \hat{\theta}_k \in \boldsymbol{\Theta} \equiv \left\{\theta : \|\theta\| \leq \mathsf{v}_{\boldsymbol{y}}^{\max}\right\} \subsetneq \mathbb{R}^2, \quad (6.5)$$

which should be as close to $\hat{\boldsymbol{x}}_{k+1|k}$ as possible. The set $\boldsymbol{\Theta} \subsetneq \mathbb{R}^2$ is natural due to the physical constraints of the speed limit.

ii) Then at time τ_{k+1}, the target arrives at state $\boldsymbol{x}_{k+1}$ and the agent arrives at $\boldsymbol{y}_{k+1}$ with the first two components specified as in (6.5). We can then repeat the same procedure in i) by setting $k \leftarrow k + 1$.

Remark 15. For (6.5) to be valid, we need to assume that the agent updates its state according to the speed $\hat{\theta}_k$ at time τ_k. The effect of the rotational dynamics are assumed negligible such that the UUV can *instantaneously* change direction for all k.

6.1.2 Loss Function

Now let us discuss the details of step i) by constructing a time-varying loss function. At time τ_k, an intuitive strategy for the agent is to pick an action $\hat{\theta}_k \in \Theta$ such that the resulting position of the agent $(y^{\mathrm{E}}_{k+1},\ y^{\mathrm{N}}_{k+1})^T$ computed as (6.5) can be as close to the target's position $(x^{\mathrm{E}}_{k+1},\ x^{\mathrm{N}}_{k+1})^T$ as possible. Namely, at time index k, we want to find a value of $\theta \in \mathbb{R}^2$ such that

$$\begin{aligned} f_k(\theta) &\equiv \frac{1}{2}\left[(x^{\mathrm{E}}_{k+1} - y^{\mathrm{E}}_{k+1})^2 + (x^{\mathrm{N}}_{k+1} - y^{\mathrm{N}}_{k+1})^2\right] \\ &= \frac{1}{2}\| \left(y^{\mathrm{E}}_k,\ y^{\mathrm{N}}_k\right)^T + (\tau_{k+1} - \tau_k)\theta - \left(x^{\mathrm{E}}_{k+1},\ x^{\mathrm{N}}_{k+1}\right)^T \|^2 \end{aligned} \tag{6.6}$$

is minimized, where the second equality is obtained by plugging in (6.5). Note that both the sampling interval $(\tau_{k+1} - \tau_k)$ and the current position of the agent $(y^{\mathrm{E}}_k,\ y^{\mathrm{N}}_k)^T$ are *known*. Unfortunately, it is not feasible for the agent to evaluate the loss function (6.6) at time τ_k, as the agent does not know the *next* position $(x^{\mathrm{E}}_{k+1},\ x^{\mathrm{N}}_{k+1})^T$ of the target at time τ_k. Even at time τ_{k+1}, the agent can only gather noisy information about $(x^{\mathrm{E}}_{k+1},\ x^{\mathrm{N}}_{k+1})^T$ through the noisy measurement (6.3). Nonetheless, the agent can instead use the approximation in (6.7) as a proxy for the true loss function (6.6):

$$\hat{f}_k(\theta) = \frac{1}{2}\| \left(y^{\mathrm{E}}_k,\ y^{\mathrm{N}}_k\right)^T + (\tau_{k+1} - \tau_k)\theta - \left(\hat{x}^{\mathrm{E}}_{k+1|k},\ \hat{x}^{\mathrm{N}}_{k+1|k}\right)^T \|^2, \tag{6.7}$$

where $\hat{x}^{\mathrm{E}}_{k+1|k}$ and $\hat{x}^{\mathrm{N}}_{k+1|k}$ are the a priori prediction for the first two components of $\boldsymbol{x}_{k+1|k}$ mentioned in step i).

The KF scheme is a natural tool to find the prediction $(\hat{x}^{\mathrm{E}}_{k+1|k},\ \hat{x}^{\mathrm{N}}_{k+1|k})^T$ for $(x^{\mathrm{E}}_{k+1},\ x^{\mathrm{N}}_{k+1})^T$. To implement the KF, we need to impose further assumptions. If the rotational dynamics are also negligible for the target as it is the case for the agent discussed in Remark 15, then the target state evolution should take the form:

$$\boldsymbol{x}_{k+1} = \begin{pmatrix} 1 & 0 & (\tau_{k+1}-\tau_k) & 0 \\ 0 & 1 & 0 & (\tau_{k+1}-\tau_k) \\ 0 & 0 & \iota_{k_1} & \iota_{k_2} \\ 0 & 0 & \iota_{k_3} & \iota_{k_4} \end{pmatrix} \boldsymbol{x}_k, \text{ such that } \|\boldsymbol{x}_{k+1}-\boldsymbol{x}_k\| \leq \mathsf{v}^{\max}_{\boldsymbol{x}}(\tau_{k+1}-\tau_k), \tag{6.8}$$

for some parameters $\iota_{k_1},\ \iota_{k_2},\ \iota_{k_3},\ \iota_{k_4}$ that manifest the change of the speed from $(\dot{x}^{\mathrm{E}}_k,\ \dot{x}^{\mathrm{N}}_k)^T \in \mathbb{R}^2$ to $(\dot{x}^{\mathrm{E}}_{k+1},\ \dot{x}^{\mathrm{N}}_{k+1})^T$. Still, realistically, the agent cannot access the exact evolution form (6.8) of the target. Hence, the following discrete-time representation of linear dynamics for the target is *assumed* by the agent:

$$\boldsymbol{x}_{k+1} = \boldsymbol{\Phi}_k \boldsymbol{x}_k + \boldsymbol{w}_k, \tag{6.9}$$

where the state transition matrix is

$$\boldsymbol{\Phi}_k = \begin{pmatrix} 1 & 0 & (\tau_{k+1} - \tau_k) & 0 \\ 0 & 1 & 0 & (\tau_{k+1} - \tau_k) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \in \mathbb{R}^{4\times 4}, \tag{6.10}$$

and $\boldsymbol{w}_k \overset{\text{i.i.d.}}{\sim} \text{Normal}(\mathbf{0}, \boldsymbol{Q}_k)$ with the following covariance matrix per [82]

$$\boldsymbol{Q}_k = \begin{pmatrix} \frac{(\tau_{k+1}-\tau_k)^3}{3} & 0 & \frac{(\tau_{k+1}-\tau_k)^2}{2} & 0 \\ 0 & \frac{(\tau_{k+1}-\tau_k)^3}{3} & 0 & \frac{(\tau_{k+1}-\tau_k)^2}{2} \\ \frac{(\tau_{k+1}-\tau_k)^2}{2} & 0 & (\tau_{k+1} - \tau_k) & 0 \\ 0 & \frac{(\tau_{k+1}-\tau_k)^2}{2} & 0 & (\tau_{k+1} - \tau_k) \end{pmatrix}. \tag{6.11}$$

Remark 16. The anticipated form of (6.10) is due to the physical law of inertia, i.e., every vehicle tends to keep its current speed (including both the direction and the magnitude). Fortunately, $\hat{\boldsymbol{x}}_{k+1|k}$ generated from KF provides a reasonable a priori estimation, even if (6.9) misspecifies (6.8).

With assumed form (6.9) of the target's motion, we may implement the KF-based estimation summarized in Algorithm 4. With the a priori estimation $\hat{\boldsymbol{x}}_{k+1|k}$ generated from Algorithm 4, we have a way to evaluate the proxy loss function (6.7). We now discuss using the SA scheme (2.1) to generate iterative estimate for the minimizer θ_k^* of the time-varying

Algorithm 4 Using KF to Predict $\boldsymbol{x}_{k+1}$ at Time τ_k

Input: $\hat{\boldsymbol{x}}_k \in \mathbb{R}^4$, $\tau_{k+1} - \tau_k \in \mathbb{R}$, $\boldsymbol{P}_k \in \mathbb{R}^{4\times 4}$, $\boldsymbol{\Phi}_k \in \mathbb{R}^{4\times 4}$ as in (6.10), $\boldsymbol{Q}_k \in \mathbb{R}^{4\times 4}$ as in (6.11), $\boldsymbol{R}_{k+1} \in \mathbb{R}^{2\times 2}$ as in (6.4), $\boldsymbol{S}_{k+1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \in \mathbb{R}^{2\times 4}$.

1: At time τ_k, predict (a priori) state estimate as $\hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_k$.
2: At time τ_k, also predict (a priori) covariance estimate as $\boldsymbol{P}_{k+1|k} = \boldsymbol{\Phi}_k \boldsymbol{P}_k \boldsymbol{\Phi}_k^T + \boldsymbol{Q}_k$.
3: At time τ_{k+1}, compute the Kalman gain as $\boldsymbol{K}_{k+1} = \boldsymbol{P}_{k+1|k}\boldsymbol{S}_{k+1}^T \left(\boldsymbol{S}_{k+1}\boldsymbol{P}_{k+1|k}\boldsymbol{S}_{k+1}^T + \boldsymbol{R}_{k+1}\right)^{-1}$.
4: At time τ_{k+1}, update (a posteriori) state estimate $\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1|k} + \boldsymbol{K}_{k+1}(\boldsymbol{z}\left(\boldsymbol{x}_{k+1}, \boldsymbol{y}_{k+1}\right) - \boldsymbol{S}_{k+1}\hat{\boldsymbol{x}}_{k+1|k})$, where the binary function $\boldsymbol{z}(\cdot,\cdot)$ is defined in (6.3).
5: At time τ_{k+1}, update (a posteriori) covariance estimate $\boldsymbol{P}_{k+1} = (\boldsymbol{I}_4 - \boldsymbol{K}_{k+1}\boldsymbol{S}_{k+1})\boldsymbol{P}_{k+1|k}$.

loss function (6.6). An unbiased estimator for

$$\boldsymbol{g}_k(\theta) \equiv \frac{\partial f_k(\theta)}{\partial \theta} = (\tau_{k+1} - \tau_k)^2 \theta + (\tau_{k+1} - \tau_k) \begin{pmatrix} y_k^{\mathrm{E}} - x_{k+1}^{\mathrm{E}} \\ y_k^{\mathrm{N}} - x_{k+1}^{\mathrm{N}} \end{pmatrix} \tag{6.12}$$

is

$$\hat{\boldsymbol{g}}_k(\theta) = (\tau_{k+1} - \tau_k)^2 \theta + (\tau_{k+1} - \tau_k) \begin{pmatrix} y_k^{\mathrm{E}} - \hat{x}_{k+1|k}^{\mathrm{E}} \\ y_k^{\mathrm{N}} - \hat{x}_{k+1|k}^{\mathrm{N}} \end{pmatrix}. \tag{6.13}$$

Remark 17. If the target is moving according to a *prescribed* trajectory, then there is no randomness in $(x_{k+1}^{\mathrm{E}},\ x_{k+1}^{\mathrm{N}})^T$. If otherwise, $\boldsymbol{g}_k(\cdot)$ here involves the randomness in $(x_{k+1}^{\mathrm{E}},\ x_{k+1}^{\mathrm{N}})^T$.

With (6.13), we can update the action of the agent using the scheme (2.6) and the corresponding constraint set $\boldsymbol{\Theta}$ is defined in (6.5). Furthermore, $\partial \boldsymbol{g}_k(\theta)/\partial\theta = (\tau_{k+1} -$

$\tau_k)^2 \boldsymbol{I}_2$ for all θ when the sampling interval $(\tau_{k+1} - \tau_k)$ is positive, and discrete sampling applies to most modern sensors.

We finish formulating the loss function for the case where there are only one agent and one target. Let us reiterate that the underlying loss function (6.6) is time-varying, as it evolves as the agent and target move with time. The proxy of the underlying loss function (6.7) to which the agent can access is stochastic as there is random noise in the measurement (6.3), and is information-based given the underlying KF-based prediction $\hat{\boldsymbol{x}}_{k+1|k}$ generated from Algorithm 4. We summarize the details in implementing step i) in Algorithm 5.

Algorithm 5 The Procedure to Generate the $\hat{\theta}_k$ Sequence For Single-Agent Single-Target Setting

---

Input: $\mathsf{v}_{\boldsymbol{y}}^{\max} \in \mathbb{R}$, $\hat{\theta}_0 \in \boldsymbol{\Theta}$, $\hat{\boldsymbol{x}}_0 \in \mathbb{R}^4$, $\boldsymbol{P}_0 \in \mathbb{R}^{4\times 4}$, $(\tau_{k+1} - \tau_k) \in \mathbb{R}$, $\boldsymbol{\Phi}_k \in \mathbb{R}^{4\times 4}$ as in (6.10), $\boldsymbol{Q}_k \in \mathbb{R}^{4\times 4}$ as in (6.11) for all $k \geq 0$, and $\boldsymbol{S}_k = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \in \mathbb{R}^{2\times 4}$, $\boldsymbol{R}_k \in \mathbb{R}^{2\times 2}$ as in (6.4) for all $k \geq 1$.

1: **for** $0 \leq k \leq K$ **do**

2: **a priori estimation** $\hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_k$ and $\boldsymbol{P}_{k+1|k} = \boldsymbol{\Phi}_k \boldsymbol{P}_k \boldsymbol{\Phi}_k^T + \boldsymbol{Q}_k$.

3: **update** $\hat{\theta}_{k+1} = \mathscr{P}_{\boldsymbol{\Theta}}[\hat{\theta}_k - a_k \hat{\boldsymbol{g}}_k(\hat{\theta}_k)]$, where $\hat{\boldsymbol{g}}_k(\cdot)$ is given in (6.13) and $\boldsymbol{\Theta} \subset \mathbb{R}^2$ given in (6.5).

Output: $\hat{\theta}_{k+1}$

4: **update** agent's position $(y_{k+1}^{\mathrm{E}},\ y_{k+1}^{\mathrm{N}})^T = (y_k^{\mathrm{E}},\ y_k^{\mathrm{N}})^T + (\tau_{k+1} - \tau_k)\hat{\theta}_k$ as in (6.5).

5: **compute** the Kalman gain $\boldsymbol{K}_{k+1} = \boldsymbol{P}_{k+1|k} \boldsymbol{S}_{k+1}^T (\boldsymbol{S}_{k+1} \boldsymbol{P}_{k+1|k} \boldsymbol{S}_{k+1}^T + \boldsymbol{R}_{k+1})^{-1}$.

6: **a posterior estimation** $\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1|k} + \boldsymbol{K}_{k+1} \left(\boldsymbol{z}(\boldsymbol{x}_{k+1}, \boldsymbol{y}_{k+1}) - \boldsymbol{S}_{k+1} \hat{\boldsymbol{x}}_{k+1|k} \right)$ for the binary function $\boldsymbol{z}(\cdot,\cdot)$ as in (6.3), and $\boldsymbol{P}_{k+1} = (\boldsymbol{I}_4 - \boldsymbol{K}_{k+1} \boldsymbol{S}_{k+1}) \boldsymbol{P}_{k+1|k}$.

7: **end for**

6.1.3 Relation With Error Bound Result in Chapter 3

Even though the tracking capability results in Chapter 3 are derived for iterates $\hat{\theta}_k$ generated from the unconstrained SA algorithm (2.1), they can be readily extended to the

constrained SA algorithm (2.6) using the non-expansivity of the projection $\mathscr{P}_{\boldsymbol{\Theta}}(\cdot)$ onto the feasible region $\boldsymbol{\Theta}$, as long as the optimizer $\boldsymbol{\theta}_k^* \in \boldsymbol{\Theta}$ for all k.

In the single-agent single-target setup, the root of true gradient function (6.12) gives the minimizer of the true loss function (6.6) $\boldsymbol{\theta}_k^* = (x_{k+1}^{\mathrm{E}} - y_k^{\mathrm{E}},\ x_{k+1}^{\mathrm{N}} - y_k^{\mathrm{N}})^T \in \mathbb{R}^2$ when $\boldsymbol{\theta}_k^*$ falls within the constraint region $\boldsymbol{\Theta}$. With the speed limit $\mathsf{v}_{\boldsymbol{x}}^{\max}$ imposed on the target in (6.8) and the speed limit $\mathsf{v}_{\boldsymbol{y}}^{\max}$ imposed on the agent in (6.5), we know that

$$\begin{aligned}
\|\boldsymbol{\theta}_{k+1}^* - \boldsymbol{\theta}_k^*\| &= \| \left(x_{k+2}^{\mathrm{E}} - y_{k+1}^{\mathrm{E}},\ x_{k+2}^{\mathrm{N}} - y_{k+1}^{\mathrm{N}}\right)^T - \left(x_{k+1}^{\mathrm{E}} - y_k^{\mathrm{E}},\ x_{k+1}^{\mathrm{N}} - y_k^{\mathrm{N}}\right)^T \| \\
&\leq \| \left(x_{k+2}^{\mathrm{E}} - x_{k+1}^{\mathrm{E}},\ x_{k+2}^{\mathrm{N}} - x_{k+1}^{\mathrm{N}}\right)^T \| + \| \left(y_{k+1}^{\mathrm{E}} - y_k^{\mathrm{E}},\ y_{k+1}^{\mathrm{N}} - y_k^{\mathrm{N}}\right)^T \| \\
&\leq \mathsf{v}_{\boldsymbol{x}}^{\max}(\tau_{k+2} - \tau_{k+1}) + \mathsf{v}_{\boldsymbol{y}}^{\max}(\tau_{k+1} - \tau_k). \qquad (6.14)
\end{aligned}$$

Given above, the assumption A.4 is met with $\mathscr{B}_k = \mathsf{v}_{\boldsymbol{x}}^{\max}(\tau_{k+2} - \tau_{k+1}) + \mathsf{v}_{\boldsymbol{y}}^{\max}(\tau_{k+1} - \tau_k)$. Also, the assumptions A.2 and A.3 are satisfied with $\mathscr{C}_k = \mathscr{L}_k = (\tau_{k+1} - \tau_k)^2$, given the gradient function as in (6.12) and the discussion in Subsection 3.2.6.

Last, we need to consider whether the assumption A.1 is met. In this single-target single-agent case, the error term defined in (3.5) becomes

$$\boldsymbol{e}_k(\boldsymbol{\theta}) = \hat{\boldsymbol{g}}_k(\boldsymbol{\theta}) - \boldsymbol{g}_k(\boldsymbol{\theta}) = (\tau_{k+1} - \tau_k) \begin{pmatrix} x_{k+1}^{\mathrm{E}} - \hat{x}_{k+1|k}^{\mathrm{E}} \\ x_{k+1}^{\mathrm{N}} - \hat{x}_{k+1|k}^{\mathrm{N}} \end{pmatrix}, \qquad (6.15)$$

where $\boldsymbol{g}_k(\cdot)$ is as (6.12) and $\hat{\boldsymbol{g}}_k(\cdot)$ is as (6.13). We will use the upper-left 2-by-2 submatrix of $\boldsymbol{P}_{k+1|k}$, which gives the covariance between $\hat{x}_{k+1|k}^{\mathrm{E}}$ and $\hat{x}_{k+1|k}^{\mathrm{N}}$, as a proxy of the

covariance matrix of $\boldsymbol{e}_k$ in (6.15). That is, we *assume* that the assumption A.1 is met with $\mathscr{M}_k$ *approximately* equaling the square root of the sum of the first two diagonal entries of $\boldsymbol{P}_{k+1|k}$.

6.1.4 Monte Carlo Simulation

We consider a time-frame $0 \leq k \leq 999$. Let the sampling frequency $(\tau_{k+1} - \tau_k) = 0.3$ seconds for $0 \leq k \leq 998$, which is the typical sample interval of the existing sensor. Assume that the target has a speed limit of $\mathsf{v}_{\boldsymbol{x}}^{\max} = 15$ meters per second, which is the average speed of the middle-class submarines. Assume that the agent has a speed limit of $\mathsf{v}_{\boldsymbol{y}}^{\max} = 30$ meters per second, which is the average speed of the top-tier submarines. Assume that the target is moving according to the following transition law:

$$\boldsymbol{x}_{k+1} = \begin{pmatrix} 1 & 0 & 0.3 & 0 \\ 0 & 1 & 0 & 0.3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \boldsymbol{x}_k + \boldsymbol{w}_k, \quad \text{for } 0 \leq k \leq 499 \text{ and } 501 \leq k \leq 999,$$

$$\text{and } \boldsymbol{x}_{k+1} = \begin{pmatrix} 1 & 0 & 0.3 & 0 \\ 0 & 1 & 0 & 0.3 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \boldsymbol{x}_k + \boldsymbol{w}_k, \quad \text{for } k = 500. \tag{6.16}$$

The above transition law is certainly *unknown* to the agent, and the agent will again use the anticipated transition matrix $\boldsymbol{\Phi}_k$ for the reason explained in Remark 16. The matrix $\boldsymbol{R}_k$ will be as in (6.4) and the matrix $\boldsymbol{Q}_k$ will be as (6.11) after plugging in the value of the sampling interval, which is 0.3 seconds.

One remaining input for implementing Algorithm 5 is $\boldsymbol{P}_0$. We assume that at $k = 0$, the target's location $\boldsymbol{x}_0$ and the agent's location $\boldsymbol{y}_0$ are uniformly-random distributed within $[-5, 5] \times [-5, 5]$, and assume that the target has an initial speed of $(\dot{x}_0^{\mathrm{E}},\ \dot{x}_0^{\mathrm{N}})^T$, whose Euclidean norm equals $\mathsf{v}_{\boldsymbol{x}}^{\max} = 15$, and a random direction uniformly sampled from $\mathrm{Uniform}(0, 2\pi)$. With such an initialization, the agent picks

$$\boldsymbol{P}_0 = \begin{pmatrix} \frac{100}{12} & 0 & 0 & 0 \\ 0 & \frac{100}{12} & 0 & 0 \\ 0 & 0 & \frac{15^2}{2} & 0 \\ 0 & 0 & 0 & \frac{15^2}{2} \end{pmatrix} = \mathrm{diag}(\frac{25}{3},\ \frac{25}{3},\ \frac{225}{2},\ \frac{225}{2}) \in \mathbb{R}^{4\times 4} \tag{6.17}$$

as an initial estimate for the covariance matrix $\mathbb{E}\left[(\hat{\boldsymbol{x}}_0 - \boldsymbol{x}_0)(\hat{\boldsymbol{x}}_0 - \boldsymbol{x}_0)^T\right]$. The first two diagonal entries in (6.17) are the variance of $\mathrm{Uniform}\,[-5, 5]$, and the lower-right 2-by-2 submatrix is given by the product of the squared of $\mathsf{v}_{\boldsymbol{x}}^{\max}$ and the 2-by-2 variance matrix of the cosine and the sine of a uniform random variable within $[0, 2\pi]$.

Relation To Results on Error Bound

Subsection 6.1.3 mentions that the value of $\mathscr{L}_k$ equals the value of $\mathscr{C}_k$ for all k, so we may pick $q_k = 2.5$ for all k as per line 11 in Algorithm 1. We then pick a gain a_k of $1.15/\mathscr{L}_k$ to implement (2.6) as per line 12 in Algorithm 1. So u_k can be computed as in (3.20), and v_k can be computed as in (3.21).

We reiterate that the error bound results (3.32) and (3.37) are obtained after averaging the performance on all the sample paths. For real-time tracking in this chapter, the agent will *not* have a chance to repeatedly rehearse the tracking mission. As a result, the tracking error bounds is *not* informative for *one* run, even though all the assumptions A.1–A.4 are satisfied (as discussed in Subsection 6.1.3).

Here, we use (3.31) "loosely" as the follows to compute a proxy of the error bound iteratively:

$$\|\hat{\boldsymbol{\theta}}_{k+1} - \boldsymbol{\theta}_k^*\| \leq \sqrt{u_k}\|\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k^*\| + \mathscr{M}_k\sqrt{v_k} + \mathscr{B}_k, \tag{6.18}$$

where $\mathscr{B}_k$ can be computed as in (6.14), and $\mathscr{M}_k$ can be approximately computed as the square root of the sum of the first two diagonal entries of $\boldsymbol{P}_{k+1|k}$ through implementing the recursive procedure described in Algorithm 5.

Simulation Results Using Algorithm 5 With Given Input

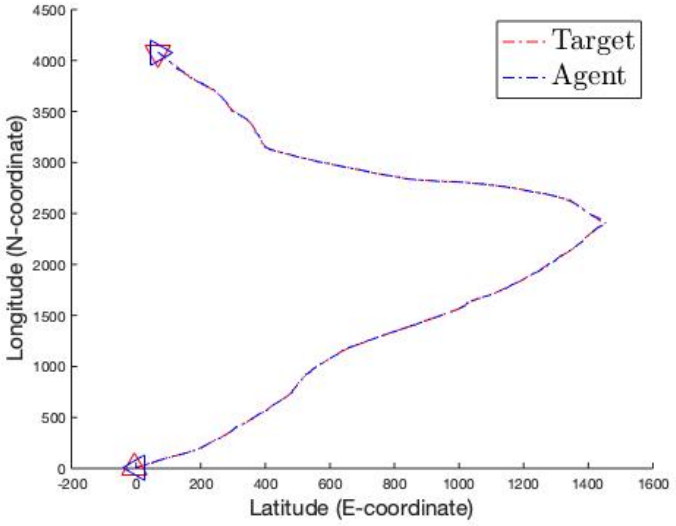
The positions of the target and the agent are plotted on the Cartesian coordinate in Figure 6.1a. The starting/ending position of the target is denoted in the red

upward/downward pointing triangle, and the initial/ending position of the agent is denoted in the blue left/right pointing triangle. The difference between the position of the target and the agent is plotted in Figure 6.1b. We also include Figure 6.2, but it is not very informative as the results in Chapter 3 is valid after averaging the performance across all sample paths.

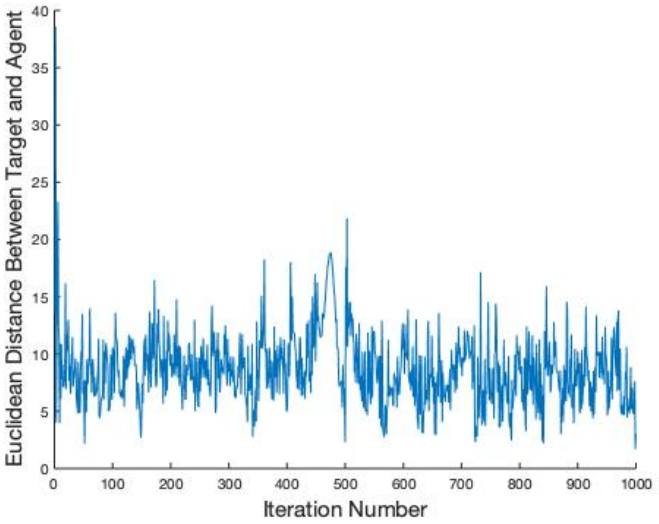
6.2 Generality: Multi-Agent Multi-Target Surveillance With Zero-Communication

We now consider the surveillance problem with I targets and J agents, for $I, J \in \mathbb{Z}$ with $1 < I \leq J$. The ith target's state and the jth agent's state at time τ_k are denoted as $\boldsymbol{x}_k^{(i)}$ and $\boldsymbol{y}_k^{(j)}$ respectively. Furthermore, we assume *no* communication between agents is allowed. Each agent should rely on local awareness and plays individually.

In the multi-agent multi-target setting with zero-communication, the objective of each agent is two-fold: one is to *track* the nearby target *if needed*, the other is to *spread* out to enlarge the collective coverage of the area of interest *if otherwise*. By "needed" we mean that the agent believes that it is closer to a certain target than any other agents. As before, the jth target is allowed to obtain noisy observations of all the targets $\boldsymbol{z}(\boldsymbol{x}_i, \boldsymbol{y}_j)$ for $i \in I$ and noisy observations of all other agents $\boldsymbol{z}(\boldsymbol{y}_{j'}, \boldsymbol{y}_j)$ for $j \in J \setminus \{j\}$, where the function $\boldsymbol{z}(\cdot, \cdot)$ is defined in (6.3).



(a) Trajectories of The Target and The Agent in *One* Simulation Run



(b) Euclidean Distance Between The Position of The Target $(x_k^{\mathrm{E}},\ x_k^{\mathrm{N}})^T$ and The Position of The Agent $(y_k^{\mathrm{E}},\ y_k^{\mathrm{N}})^T$.

Figure 6.1: A Demonstration of Implementing Algorithm 5 Using the Inputs Described in This Subsection

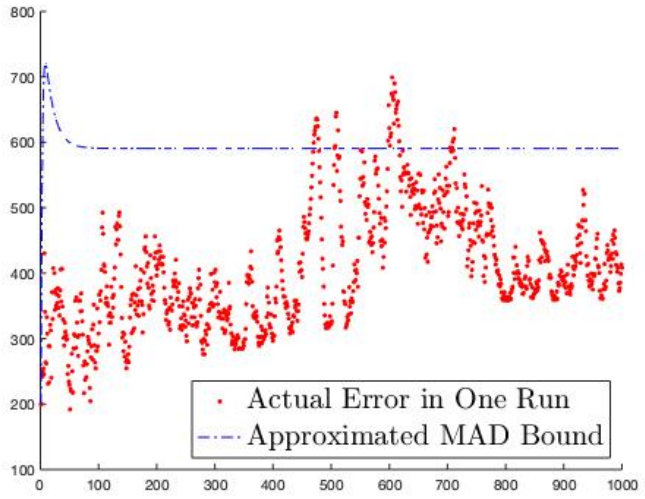


Figure 6.2: Actual Error $\|\hat{\theta}_k - \theta_k^*\|$ for One Simulation Run and the "Loose" Bound Computed Per (6.18)

6.2.1 Loss Function

Let us explain the loss function addressing the objective of "actively-tracking if needed and actively-spreading if otherwise" that applies to every agent. If all the targets are *equally* important, the ensemble of the agents is likely to distribute

$$j^* \leq \lfloor J/I \rfloor \tag{6.19}$$

agent(s) to track each target. Usually, $j^* = 1$. Algorithm 6 describes the procedure to assign "actively-tracking" and "actively-spreading" agents.

At time τ_k, we assign what we will call the "actively-tracking" agents and the "actively-spreading" agents from this point on using Algorithm 6. Specifically, the output $T_k^{(i)}$ of Algorithm 6 contains the indices of the agents that are expected to track target i

for $1 \leq i \leq I$, and the output S_k (if nonempty) contains the indices of the agents that are expected to spread out as much as possible.

Algorithm 6 Assigning Actively-Tracking and Actively-Spreading Agents At Time τ_k

Input: the distance matrix $\boldsymbol{D}_{k+1}$ with the (i,j)-entry being $(x_{k+1}^{(i),\mathrm{E}} - y_{k+1}^{(j),\mathrm{E}})^2 + (x_{k+1}^{(i),\mathrm{N}} - y_{k+1}^{(j),\mathrm{N}})^2$ for $1 \leq i \leq I$ and $1 \leq j \leq J$, and the desired number j^* of agents to keep track of a single target satisfying (6.19).

1: **initialize** the search set $S_k = \{1, \cdots, J\}$.
2: **find** a set $T_k^{(1)} \subseteq S_k$ that contains the indices of the columns that have the j^* smallest entries within the first row of $\boldsymbol{D}_{k+1}$. ▷ If there exist equal rankings, just pick any set such that cardinality of $T_k^{(1)}$ is j^*.
3: **for** (**do** $2 \leq i \leq I$)
4: **update** $S_k \leftarrow S_k \setminus T_k^{(i-1)}$.
5: **find** a set $T_k^{(i)} \subseteq S_k$ that contains the indices of the column that have the j^* smallest entries in the ith row of $\boldsymbol{D}_{k+1}$.
6: **end for**
7: **update** the search set $S_k \leftarrow S_k \setminus T_k^{(I)}$.

Output: the sets $T_k^{(1)}, \cdots, T_k^{(I)}$ and the set S_k.

If S_k is nonempty, then the jth agent for $j \in S_k$ is not expected to actively track any of the targets and should spread out as much as possible to maximize the coverage area. There are many ways to quantify "spreading" and we adopt the strategy proposed by [66]. The notion of Voronoi cell (a.k.a. Thiessen polygon) is used. Let $\boldsymbol{\zeta}$ be any canonical point in $\mathbb{R}^2$, then the Voronoi cell $V_k^{(j)} \subsetneq \mathbb{R}^2$ within which the jth agent locates is constructed in a way such that for any canonical point $\boldsymbol{\zeta} \in V_k^{(j)}$, the distance between $\boldsymbol{\zeta}$ and the jth agent's position is strictly smaller than the distance between $\boldsymbol{\zeta}$ and the position of any other agent at time τ_k. The required input (which is a proper subspace of the two-dimensional Euclidean space) to compute the Voronoi cells $V_k^{(j)}$ for $1 \leq j \leq J$ is the convex hull of all the agent's positions $(y_{k+1}^{(j),\mathrm{E}},\ y_{k+1}^{(j),\mathrm{N}})^T$ for all $1 \leq j \leq J$. Given that directly minimizing

(6.21) is difficult, an intuitive alternative is to let the "actively-spreading" agents reach the center of the mass of $V_k^{(j)}$ for $j \in S_k$, which can be computed as:

$$\boldsymbol{c}_k^{(j)} = \frac{\int_{\zeta \in V_k^{(j)}} \boldsymbol{\zeta} d\boldsymbol{\zeta}}{\text{Area of } V_k^{(j)}}. \tag{6.20}$$

With the notion of Voronoi cell, [66] minimizes the following loss function

$$\begin{aligned} &\sum_{j \in S_k} \int_{V_k^{(j)}} \|\boldsymbol{\zeta} - (y_{k+1}^{(j),\mathrm{E}},\ y_{k+1}^{(j),\mathrm{N}})^T\|^2 d\boldsymbol{\zeta} \\ &\quad = \sum_{j \in S_k} \int_{V_k^{(j)}} \|(y_k^{(j),\mathrm{E}},\ y_k^{(j),\mathrm{N}})^T + (\tau_{k+1} - \tau_k)\boldsymbol{\theta}^{(j)} - \boldsymbol{\zeta}\|^2 d\boldsymbol{\zeta} \end{aligned} \tag{6.21}$$

w.r.t. the actions $\boldsymbol{\theta}^{(j)} \in \boldsymbol{\Theta} \subsetneq \mathbb{R}^2$ for $j \in S_k$ and for $\boldsymbol{\Theta}$ defined as in (6.5). In (6.21), $\boldsymbol{\zeta}$ is any canonical point in $\mathbb{R}^2$, and $V_k^{(j)} \subsetneq \mathbb{R}^2$ denotes the Voronoi cell [21, Sect. 8.11] within which the jth agent's locates.

To achieve the goal of "actively-tracking if needed and actively-spreading if otherwise," every agent strives to minimize the following loss function

$$\begin{aligned} f_k(\boldsymbol{\theta}) &= \frac{1}{2}\sum_{i=1}^{I}\sum_{j\in T_k^{(i)}}\left((x_{k+1}^{(i),\mathrm{E}}-y_{k+1}^{(j),\mathrm{E}})^2+(x_{k+1}^{(i),\mathrm{N}}-y_{k+1}^{(j),\mathrm{N}})^2\right) \\ &\quad+\frac{1}{2}\sum_{j\in S_k}\|\begin{pmatrix}y_{k+1}^{(j),\mathrm{E}}, & y_{k+1}^{(j),\mathrm{N}}\end{pmatrix}^T-\boldsymbol{c}_k^{(j)}\|^2 \\ &=\frac{1}{2}\sum_{i=1}^{I}\sum_{j\in T_k^{(i)}}\|(y_k^{(j),\mathrm{E}}-x_{k+1}^{(i),\mathrm{E}},\ y_k^{(j),\mathrm{N}}-x_{k+1}^{(i),\mathrm{N}})^T+(\tau_{k+1}-\tau_k)\boldsymbol{\theta}^{(j)}\|^2 \\ &\quad+\frac{1}{2}\sum_{j\in S_k}\|(y_k^{(j),\mathrm{E}},\ y_k^{(j),\mathrm{N}})^T+(\tau_{k+1}-\tau_k)\boldsymbol{\theta}^{(j)}-\boldsymbol{c}_k^{(j)}\|^2 \end{aligned} \tag{6.22}$$

w.r.t. $\boldsymbol{\theta}\in\boldsymbol{\Theta}^J\subsetneq\mathbb{R}^{2J}$, where $\boldsymbol{\theta}$ is the concatenation of $\boldsymbol{\theta}^{(j)}\in\boldsymbol{\Theta}\subsetneq\mathbb{R}^2$ for all $1\le j\le J$.

Nonetheless, under the zero-communication setting, there exists no commander in chief who can dispatch the corresponding actions $\boldsymbol{\theta}_k\in\mathbb{R}^{2J}$ to all the agents using the information from $f_k(\cdot)$. Consequently, the jth agent only gets to update its action $\boldsymbol{\theta}_k^{(j)}\in\boldsymbol{\Theta}$

by minimizing the following loss function

$$
\begin{aligned}
& f_k^{(j)}(\boldsymbol{\theta}^{(j)}) \\
& \quad = \frac{1}{2}\sum_{i=1}^{I}\left\{\mathbb{I}_{\{j\in T_k^{(i)}\}}\times[(x_{k+1}^{(i),\mathrm{E}}-y_{k+1}^{(j),\mathrm{E}})^2+(x_{k+1}^{(i),\mathrm{N}}-y_{k+1}^{(j),\mathrm{N}})^2]\right\} \\
& \qquad + \frac{1}{2}\mathbb{I}_{\{j\in S_k\}}\times\|(y_{k+1}^{(j),\mathrm{E}},\ y_{k+1}^{(j),\mathrm{N}})^T-\boldsymbol{c}_k^{(j)}\|^2 \\
& \quad = \begin{cases} \dfrac{1}{2}\|(y_k^{(j),\mathrm{E}}-x_{k+1}^{(i),\mathrm{E}},\ y_k^{(j),\mathrm{N}}-x_{k+1}^{(i),\mathrm{N}})^T+(\tau_{k+1}-\tau_k)\boldsymbol{\theta}^{(j)}\|^2, \\ \qquad\qquad \text{if } j\in T_k^{(i)} \text{ for } \textit{some } 1\le i\le I, \\ \dfrac{1}{2}\|(y_k^{(j),\mathrm{E}},\ y_k^{(j),\mathrm{N}})^T-\boldsymbol{c}_k^{(j)}+(\tau_{k+1}-\tau_k)\boldsymbol{\theta}^{(j)}\|^2, \text{ if } j\in S_k, \end{cases}
\end{aligned}
\tag{6.23}
$$

w.r.t. $\boldsymbol{\theta}^{(j)}\in\mathbb{R}^2$. According to Algorithm6, $T_k^{(1)},\cdots,T_k^{(I)},S_k$ are mutually exclusive.

Of course, at time τ_k, the agent j does *not* have $\boldsymbol{x}_{k+1}^{(i)}$ for $1\le i\le I$ and $\boldsymbol{y}_{k+1}^{(j')}$ for $j'\neq j$ to determine $T_k^{(i)}$ and S_k for $1\le i\le I$ using Algorithm 6 and to compute the Voronoi cells $V_k^{(j)}$ and the centers $\boldsymbol{c}_k^{(j)}$ for $1\le j\le J$. Similar to the rationale behind substituting (6.6) for (6.7), the agent j can use (6.24) as a proxy of (6.23):

$$
\hat{f}_k^{(j)}(\boldsymbol{\theta}^{(j)}) = \begin{cases} \dfrac{1}{2}\|(y_k^{(j),\mathrm{E}}-\hat{x}_{k+1|k}^{(i),\mathrm{E},(j)},\ y_k^{(j),\mathrm{N}}-\hat{x}_{k+1|k}^{(i),\mathrm{N},(j)})^T+(\tau_{k+1}-\tau_k)\boldsymbol{\theta}^{(j)}\|^2, \text{ if } j\in\hat{T}_k^{(i,j)}, \\ \dfrac{1}{2}\|(y_k^{(j),\mathrm{E}},\ y_k^{(j),\mathrm{N}})^T+(\tau_{k+1}-\tau_k)\boldsymbol{\theta}^{(j)}-\hat{\boldsymbol{c}}_k^{(j,j)}\|^2, \text{ if } j\in\hat{S}_k^{(j)}. \end{cases}
\tag{6.24}
$$

where all the relevant computations arising in (6.23) are executed using the current state of the jth agent $\boldsymbol{y}_k$, and the a priori approximation $\hat{\boldsymbol{x}}_{k+1|k}^{(i,j)}$ for $1\le i\le I$ and $\hat{\boldsymbol{y}}_{k+1|k}^{(j',j)}$ for $j'\neq j$ based on the information available to agent j, including (1) finding $\hat{T}_k^{(i,j)}$ and $\hat{S}_k^{(j)}$

through implementing Algorithm 6 and (2) generating Voronoi cells $\hat{V}_k^{(j,j)}$ using the built-in MATLAB function `voronoi(·)` and computing the centers $\hat{\boldsymbol{c}}_k^{(j,j)}$ using `polygem(·)`. $\hat{T}_k^{(i,j)}$, $\hat{S}_k^{(j)}$, $\hat{V}_k^{(j,j)}$, and $\hat{\boldsymbol{c}}_k^{(j,j)}$ in (6.24) represent the estimation of $T_k^{(i)}$, S_k, $V_k^{(j)}$, and $\boldsymbol{c}_k^{(j)}$ appearing in (6.23) based on the estimation obtained by the jth agent.

A ready estimator, which may be *biased* due to the potential inconsistency between $T_k^{(i)}$ ($\hat{T}_k^{(i,j)}$) and S_k ($\hat{S}_k^{(j)}$), for

$$
\begin{aligned}
&\boldsymbol{g}_k^{(j)}(\boldsymbol{\theta}^{(j)}) \\
&\quad \equiv \frac{\partial f_k^{(j)}(\boldsymbol{\theta}^{(j)})}{\partial \boldsymbol{\theta}^{(j)}} \\
&\quad = \begin{cases} (\tau_{k+1}-\tau_k)^2\boldsymbol{\theta}^{(j)} + (\tau_{k+1}-\tau_k)\begin{pmatrix} y_k^{(j),\mathrm{E}} - x_{k+1}^{(i),\mathrm{E}} \\ y_k^{(j),\mathrm{N}} - x_{k+1}^{(i),\mathrm{N}} \end{pmatrix}, \text{ if } j \in T_k^{(i)} \text{ for some } 1 \le i \le I, \\ (\tau_{k+1}-\tau_k)^2\boldsymbol{\theta}^{(j)} + (\tau_{k+1}-\tau_k)\begin{pmatrix} y_k^{(j),\mathrm{E}} \\ y_k^{(j),\mathrm{N}} \end{pmatrix} - (\tau_{k+1}-\tau_k)\boldsymbol{c}_k^{(j)}, \text{ if } j \in S_k. \end{cases}
\end{aligned}
\tag{6.25}
$$

is

$$
\begin{aligned}
&\hat{\boldsymbol{g}}_k^{(j)}(\theta^{(j)}) \\
&\quad \equiv \frac{\partial \hat{f}_k^{(j)}(\theta^{(j)})}{\partial \theta^{(j)}} \\
&\quad = \begin{cases} (\tau_{k+1}-\tau_k)^2\theta^{(j)} + (\tau_{k+1}-\tau_k)\begin{pmatrix} y_k^{(j),\mathrm{E}} - \hat{x}_{k+1|k}^{(i),\mathrm{E},(j)} \\ y_k^{(j),\mathrm{N}} - \hat{x}_{k+1|k}^{(i),\mathrm{N},(j)} \end{pmatrix}, \\ \qquad \text{if } j \in \hat{T}_k^{(i,j)} \text{ for some } 1 \le i \le I, \\ (\tau_{k+1}-\tau_k)^2\theta^{(j)} + (\tau_{k+1}-\tau_k)\begin{pmatrix} y_k^{(j),\mathrm{E}} \\ y_k^{(j),\mathrm{N}} \end{pmatrix} - (\tau_{k+1}-\tau_k)\hat{\boldsymbol{c}}_k^{(j,j)}, \text{ if } j \in \hat{S}_k^{(j)}. \end{cases}
\end{aligned}
\tag{6.26}
$$

A natural strategy to decide the action $\theta^{(j)}$ of the agent j is the truncated SA algorithm (2.6) with $\boldsymbol{\Theta}$ as in (6.5). Furthermore, $\partial \boldsymbol{g}_k^{(j)}(\theta)/\partial\theta = (\tau_{k+1}-\tau_k)^2\boldsymbol{I}_2$ for all θ when the sampling interval $\tau_{k+1}-\tau_k$ is positive. Hence, all the discussion in Subsection 6.1.3 regarding the loss function $f_k(\cdot)$ in (6.6) is applicable for the loss function $f_k^{(j)}(\cdot)$ in (6.23) for all $1 \le j \le J$.

We finish stating the loss function for the general case where there are multiple agents and multiple targets. Again the underlying loss function (6.23) for agent j is time-varying, and it only gets access to the noisy evaluation $\boldsymbol{z}(\boldsymbol{x}_i, \boldsymbol{y}_j)$ for $1 \le i \le I$ and $\boldsymbol{z}(\boldsymbol{y}_{j'}, \boldsymbol{y}_j)$ for $j' \ne j$.

Algorithm 7 The Procedure to Generate $\hat{\theta}_k^{(j)}$ Sequence For jth Agent In Multi-Agent Multi-Target Setting

Input: $\mathsf{v}_{\boldsymbol{y}}^{\max} \in \mathbb{R}$, $\hat{\theta}_0^{(j)} \in \boldsymbol{\Theta}$, $(\tau_{k+1} - \tau_k) \in \mathbb{R}$ for all $k \geq 0$, $\hat{\boldsymbol{x}}_0^{(i,j)} \in \mathbb{R}^4$ and $\boldsymbol{P}_0^{(i,j)} \in \mathbb{R}^{4\times 4}$ for all $1 \leq i \leq I$, $\hat{\boldsymbol{y}}_0^{(j',j)} \in \mathbb{R}^4$ and $\tilde{\boldsymbol{P}}_0^{(j',j)}$ for all $j' \neq j$, $\boldsymbol{\Phi}_k \in \mathbb{R}^{4\times 4}$ as in as in (6.10), $\boldsymbol{Q}_k \in \mathbb{R}^{4\times 4}$ as in (6.11) for all $k \geq 0$, and $\boldsymbol{S}_k = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \in \mathbb{R}^{2\times 4}$, $\boldsymbol{R}_k \in \mathbb{R}^{2\times 2}$ as in (6.4) for all $k \geq 1$.

1: **for** $0 \leq k \leq K$ **do**
2: **for** $1 \leq i \leq I$ **do**
3: **a priori estimation** $\hat{\boldsymbol{x}}_{k+1|k}^{(i,j)} = \boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_k^{(i,j)}$ and $\boldsymbol{P}_{k+1|k}^{(i,j)} = \boldsymbol{\Phi}_k \boldsymbol{P}_k^{(i,j)} \boldsymbol{\Phi}_k^T + \boldsymbol{Q}_k$.
4: **end for**
5: **for** $1 \leq j' \leq J$ and $j' \neq j'$ **do**
6: **a priori estimation** $\hat{\boldsymbol{y}}_{k+1|k}^{(j',j)} = \boldsymbol{\Phi}_k \hat{\boldsymbol{y}}_k^{(j',j)}$ and $\tilde{\boldsymbol{P}}_{k+1|k}^{(j',j)} = \boldsymbol{\Phi}_k \tilde{\boldsymbol{P}}_k^{(j',j)} \boldsymbol{\Phi}_k^T + \boldsymbol{Q}_k$.
7: **end for**
8: **generate** $\hat{T}_k^{(i,j)}$ and $\hat{S}_k^{(j)}$ (via Algorithm 6) and **compute** $\hat{V}_k^{(j,j)}$ and $\boldsymbol{c}_k^{(j,j)}$ using $\hat{\boldsymbol{x}}_{k+1|k}^{(i,j)}$ for $1 \leq i \leq I$ and $\hat{\boldsymbol{y}}_{k+1|k}^{(j',j)}$ for $j' \neq j$ as input.
9: **update** $\hat{\theta}_{k+1}^{(j)} = \mathscr{P}_{\boldsymbol{\Theta}}[\hat{\theta}_k^{(j)} - a_k^{(j)} \hat{\boldsymbol{g}}_k^{(j)}(\hat{\theta}_k^{(j)})]$ where $\hat{\boldsymbol{g}}_k^{(j)}(\cdot)$ is given in (6.26).

Output: $\hat{\theta}_{k+1}^{(j)}$

10: **update** the jth agent's position $(y_{k+1}^{(j),\mathrm{E}},\ y_{k+1}^{(j),\mathrm{N}})^T = (y_k^{(j),\mathrm{E}},\ y_k^{(j),\mathrm{N}})^T + (\tau_{k+1} - \tau_k)\hat{\theta}_k^{(j)}$.
11: **for** $1 \leq i \leq I$ **do**
12: **compute** the Kalman gain $\boldsymbol{K}_{k+1}^{(i,j)} = \boldsymbol{P}_{k+1|k}^{(i,j)} \boldsymbol{S}_{k+1}^T (\boldsymbol{S}_{k+1} \boldsymbol{P}_{k+1|k}^{(i,j)} \boldsymbol{S}_{k+1}^T + \boldsymbol{R}_{k+1})^{-1}$.
13: **a posterior estimation** $\hat{\boldsymbol{x}}_{k+1}^{(i,j)} = \hat{\boldsymbol{x}}_{k+1|k}^{(i,j)} + \boldsymbol{K}_{k+1}^{(i,j)} (\boldsymbol{z}(\boldsymbol{x}_{k+1}^{(i)}, \boldsymbol{y}_{k+1}^{(j)}) - \boldsymbol{S}_{k+1} \hat{\boldsymbol{x}}_{k+1|k}^{(i,j)})$ for the binary function $\boldsymbol{z}(\cdot,\cdot)$ as in (6.3), and $\boldsymbol{P}_{k+1}^{(i,j)} = (\boldsymbol{I}_4 - \boldsymbol{K}_{k+1}^{(i,j)} \boldsymbol{H}_{k+1}) \boldsymbol{P}_{k+1|k}^{(i,j)}$.
14: **end for**
15: **for** $1 \leq j' \leq J$ and $j' \neq j$ **do**
16: **compute** the Kalman gain $\tilde{\boldsymbol{K}}_{k+1}^{(j',j)} = \tilde{\boldsymbol{P}}_{k+1|k}^{(j',j)} \boldsymbol{S}_{k+1}^T (\boldsymbol{S}_{k+1} \tilde{\boldsymbol{P}}_{k+1|k}^{(j',j)} \boldsymbol{S}_{k+1}^T + \boldsymbol{R}_{k+1})^{-1}$.
17: **a posterior estimation** $\hat{\boldsymbol{y}}_{k+1}^{(j',j)} = \hat{\boldsymbol{y}}_{k+1|k}^{(j',j)} + \tilde{\boldsymbol{K}}_{k+1}^{(j',j)} (\boldsymbol{z}(\boldsymbol{y}_{k+1}^{(j')}, \boldsymbol{y}_{k+1}^{(j)}) - \boldsymbol{S}_{k+1} \hat{\boldsymbol{y}}_{k+1|k}^{(j',j)})$ for the binary function $\boldsymbol{z}(\cdot,\cdot)$ as in (6.3), and $\tilde{\boldsymbol{P}}_{k+1}^{(j',j)} = (\boldsymbol{I}_4 - \tilde{\boldsymbol{K}}_{k+1}^{(j',j)} \boldsymbol{H}_{k+1}) \tilde{\boldsymbol{P}}_{k+1|k}^{(j',j)}$.
18: **end for**
19: **end for**

Let us summarize the estimation procedure in Algorithm 7. Lines 2—7 compute the a priori estimate for the states of all the targets and all the other agents. Lines 8 decides whether the jth agent is "actively-tracking" (i.e., $j \in \hat{T}_k^{(i,j)}$ for some $1 \leq i \leq I$) or is "actively-spreading" (i.e., $j \in \hat{S}_k^{(j)}$). Line 9 is to pick a decision $\hat{\boldsymbol{\theta}}_k^{(j)}$ using the truncated SA scheme (2.6). Then line 10 is to update the jth agent's position according to $\hat{\boldsymbol{\theta}}_k^{(j)}$ and (6.5). Lines 11—18 update the a posterior estimate for the states of all the targets and all the other agents.

6.2.2 Monte Carlo Simulation

This subsection will use the same initialization as Subsection 6.1.4, except that $\mathsf{v}_{\boldsymbol{y}}^{\max}$ becomes the same as $\mathsf{v}_{\boldsymbol{x}}^{\max} = 15$ meters per seconds. This change is made in the hope that the requirement on the agent's UUV speed in the multi-agent setting with the joint effort with an ensemble of agents will not be as stringent as the requirement in the single-agent setting.

For graphical illustration, we use $J = 4$ agents to track $I = 2$ targets, and we pick j^* to be 1 per (6.19). The positions of two targets and four agents from $0 \leq k \leq 999$ are plotted on the two-dimensional plane in Figure 6.3. The staring/ending positions of the first target are denoted in the red upward/downward pointing triangles, and those of the second target are denoted in black. The staring/ending positions of four agents is denoted in the left/right pointing triangles, and they are in blue, magenta, yellow, and cyan respectively. We can see that two agents are "actively-tracking" as they follow the two targets closely,

and two agents are "actively-spreading" as they are randomly moving to somewhere in the middle of the simulation runs and end up in the positions that are not close to any of the targets. This is what an ensemble of agents would look like as they are all trying to achieve "actively-tracking if needed and actively-spreading if otherwise."

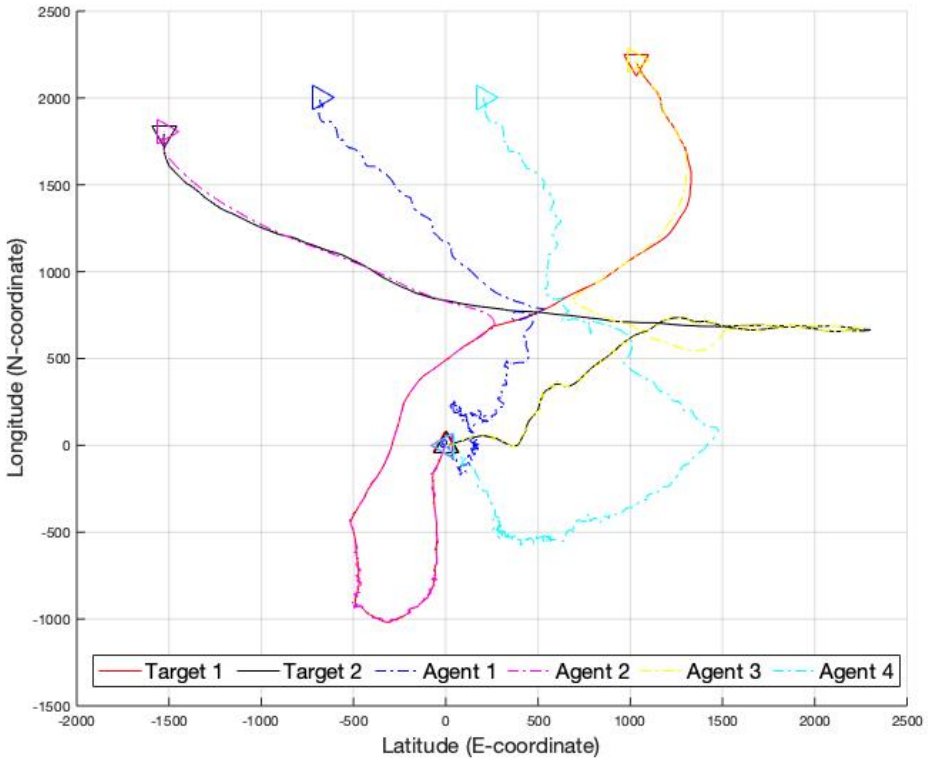


Figure 6.3: Trajectories of Two Targets and Four Agents in *One* Simulation Run.

6.3 Further Discussion

This numerical chapter presents an investigation on the performance of the general SA algorithm (2.6) in this multi-agent multi-target setting by simulating their dynamics.

Here, several agents in UUV need to perform a surveillance task within a certain area of interest such that any "intruders" in the coverage area can be tracked. Subsections 6.1.2 and 6.2.1 formulate the surveillance problem as a stochastic optimization problem under time-varying setting. Algorithms 5 and 7 demonstrate how the SA algorithms with non-decaying gain is applied to conduct the time-varying SO task. Besides, the numerical results in Subsections 6.1.4 and 6.2.2 partly manifest the error bound results in Chapter 3. The data-dependent gain-tuning strategy proposed in Chapter 5 can also be applied in the multi-agent application. Nonetheless, there are many other subtitles we avoid on purpose to present a clean story. The real-world application may be different from the procedure described in this chapter due to various factors, e.g., the agent can collect a four-dimensional (as opposed to the two-dimensional reading (6.3)) reading including the speed of the agent using Doppler radar, or the number of agents J is smaller than the number of targets I, and so on. We mention a few of them that need to be dealt with in real-world tracking problems here.

6.3.1 Detection Model

In both Section 6.1 and Section 6.2, we assume that each agent has an infinite detection range, i.e., each agent can collect noisy measurements (6.3) between itself and any other object (either a target or an agent), so as to present the loss functions (6.6) and (6.23) concisely. In reality, each agent is only allowed to gather readings (6.3) from its nearby surroundings and use these readings to estimate the position of the detectable objects. That

is, the agent's sensor can only detect its surroundings up to a certain distance denoted as μ. If the object falls within the μ-neighborhood of the agent, it can be detected by the agent; otherwise, it is practically "invisible" to the agent. The typical value of the detection range μ can be as small as 30 meters or as large as 1600 meters.

In addition to the detection range, the detection accuracy is also an important factor that affects the loss function formulation. Intuitively, the further the object is away from the agent, the less informative is the noisy measurement between the object and the agent itself, even *if* the target stays within the detection range of the agent. For example, we can borrow the idea in [57] to model the detection accuracy, which is measured by the magnitude of the covariance matrix $\boldsymbol{R}_k$ of the measurement noise $\boldsymbol{v}_k$ arising in (6.3). By taking the detection accuracy into consideration, the diagonal entries of $\boldsymbol{R}_k$ can be a non-decreasing function of the actual distance between the agent and the object (either a target or another agent). The domain of this non-decreasing function would be $[0, \mu]$ for the detection range of μ, and the function form can be linear, exponential, and so on.

6.3.2 Communication Within Range

Another factor comes into play when the detection range in Subsection 6.3.1 is taken into account. If two agents are close to each other, i.e., one falls within the detection range of the other, then the communication between these two agents is generally allowed in reality. With the allowable communication within range at time τ_k, two neighboring agents can exchange information, including their estimations for the states of all the targets and

all other agents. The neighboring agents can rely on their mutual awareness and plan in a local team (as opposed to individually in Section 6.2) as a fully connected ensemble.

Let j_1 and j_2 be the indices of two neighboring agents at time τ_k. They can share the following with each other:

$$\hat{\boldsymbol{x}}_k^{(i,j_1)}, \hat{\boldsymbol{x}}_k^{(i,j_2)} \text{ for all } 1 \leq i \leq I, \text{ and } \hat{\boldsymbol{y}}_k^{(j',j_1)}, \hat{\boldsymbol{y}}_k^{(j',j_2)} \text{ for all } j' \in \{1, \cdots, J\} \setminus \{j_1, j_2\}. \tag{6.27}$$

It is natural to take advantage of the shared information (6.27) to jointly improve their estimations. For example, to obtain a better estimate of $\hat{\boldsymbol{x}}_k^{(i)}$, the two neighboring agents can take a *weighted* average between $\hat{\boldsymbol{x}}_k^{(i,j_1)}$ and $\hat{\boldsymbol{x}}_k^{(i,j_2)}$. The weighted average is proportional to $(\boldsymbol{P}_k^{(i,j_1)})^{-1}\hat{\boldsymbol{x}}_k^{(i,j_1)} + (\boldsymbol{P}_k^{(i,j_2)})^{-1}\hat{\boldsymbol{x}}_k^{(i,j_2)}$ where the $(\boldsymbol{P}_k^{(i,j_1)})^{-1}$ measures the degree of confidence in using $\hat{\boldsymbol{x}}_k^{(i,j_1)}$ as an estimate for $\hat{\boldsymbol{x}}_k^{(i)}$, and $(\boldsymbol{P}_k^{(i,j_2)})^{-1}$ measures the degree of confidence of using $\hat{\boldsymbol{x}}_k^{(i,j_2)}$ as an estimate for $\hat{\boldsymbol{x}}_k^{(i)}$. Similarly, agent j_1 and j_2 can jointly improve their estimate for $\hat{\boldsymbol{y}}_k^{(j')}$ for $j' \in \{1, \cdots, J\} \setminus \{j_1, j_2\}$.

Of course, the weighted average of multiple agents' information is also possible when they are within the detection range of each other. We haven't implemented this weighted average idea in our simulation study, as the notion of detection range in Subsection 6.3.1 is not considered in this numerical study. It is also "dangerous" in some situations where agents need to stay hidden–minimal (or no) transmissions are preferred.

Chapter 7

Summary and Possible Future Work

The thesis considers the general stochastic approximation setup, under a practical situation where the scalar-valued objective function $f_k(\cdot)$ or the vector-valued function $\boldsymbol{g}_k(\cdot)$ may be perpetually time-varying. The method we investigated is the general SA recursive schemes (2.1) with non-decaying gains. The time-varying problem setting and SA framework have presented several new issues, both theoretical and practical.

Chapter 3 develops bounds for both MAD and RMS: the unconditional version is obtained by averaging all possible sample paths, and the conditional version is gathered by observing actual noisy gradient information. Both error bounds are computable as long as we have access to the noise level and the Hessian matrix of the underlying loss function. Note that our quantification of tracking capability within finite-iterations of the non-diminishing gain SA algorithm is in terms of a probabilistically *computable* error bound, which may also apply to the general *nonlinear* SA literature. These two

characteristics make our work different from [34] which focuses on linear models, and [117] which provides big-O bounds. Moreover, to the best of our knowledge, there are no existing approaches in estimation theory that solve a sequence of the time-varying problem, under only Assumption A.4 (the expected distance between two consecutive optima are bounded from above) without any further stringent state evolution assumption. A.4 is a fairly modest assumption on the evolution of the underlying time-varying parameter to be identified: the average distance between two consecutive minimizers $\|\theta_{k+1}^* - \theta_k^*\|$ is bounded uniformly across k. Finally, as a consequence of the MAD bound, we can characterize the stability of the SA algorithm in response to the drift $\{\theta_k^*\}$ in terms of determining the allowable region for the non-diminishing gain a, which embraces many more general SA algorithms including the special case of SGD discussed in [123]. In short, the tracking performance of non-diminishing gain SA algorithms is guaranteed by a computable bound on MAD, which is useful in *finite-sample* performance.

To supplement the tracking capability discussed in Chapter 3, Chapter 4 focuses on the concentration behavior in terms of the probabilistic bound of the recursive estimates generated from the constant-gain SGD algorithm over a finite time frame. The weak convergence limit of a suitably interpolated sequence of the iterates is shown to follow the trajectory of a non-autonomous ordinary differential equation, and the discussion there applies to constrained optimization in Section 4.1. The weak convergence limit is taken w.r.t. the constant gain a. It should be interpreted that for some nonzero constant gain a, which needs not to go to zero, the continuation of $\hat{\theta}_k$ will stay close in the sense of weak

limit to a non-autonomous ODE when the underlying data change with time on a scale that is commensurate with what is determined by the gain. To make the bound of the probability for the event that $\hat{\theta}_k$ deviates from θ_k^* computable, Section 4.2 imposes further assumptions and utilizes the formula for variation of parameters. The probabilistic bound there provides a general sense of the likelihood of $\hat{\theta}_k$ staying close to θ_k^* for a constant gain a under certain conditions. Note that the upper bound for the probability of the iterates deviating from the target is valid for all time, which is useful for finite-sample analysis.

Even though Chapter 3 develops a gain tuning strategy based upon the MAD bound, the strategy is derived after averaging out all possible sample paths of the random sequence $\{\theta_k^*\}$. Even though Chapter 4 discusses the weak convergence limit and a bound of the event that $\hat{\theta}_k$ deviates from θ_k^*, it only characterizes the small probability of the rare event of θ_k^* deviates from θ_k^* beyond a certain threshold. Both of these are probabilistic arguments and may not provide much help in tuning the non-decaying gain in practical implementations. In reality, we hope to detect the changes in $\{\theta_k^*\}$ as promptly and accurately as possible. Moreover, we have to deal with the situation where the Hessian and error information that governs the MAD bound are *unavailable*. These two reasons motivate us to direct our attention to a data-dependent gain tuning strategy. Taking advantage of observable data helps improve the tracking performance on each specific sample-path. Thus, Section 5.1 develops a change detection strategy, using the test statistic in the multivariate Behrens–Fisher problem, although the detection relies on the approximately normal distribution of the estimates $\{\hat{\theta}_k\}$ when it reaches steady-state phase

and oscillates around θ_k^*. We, unfortunately, cannot provide exact type-I and type-II errors for such a test. Nonetheless, the detection scheme does help to detect regime change robustly and avoid the burden of estimating Hessian and noise level adaptively. Based on the change detection testing in Sections 5.1 and 5.2, we continue to develop a gain adaptation strategy to adaptively adjust the gain sequence by detecting whether a jump has occurred or not. To perform better with each sample-path, we have to adjust the gain sequence adaptively based on the given data stream. Here, we handle the issues of the Hessian and noise levels being unknown by using simultaneous perturbation methods, which is efficient and inexpensive.

In a nutshell, this work partly answers the questions "what is the estimate for the dynamical system θ_k^*" and "how much we can trust $\hat{\theta}_k$ as an estimate for θ_k^*." To the best of our knowledge, there are no existing approaches in estimation theory that solve a sequence of time-varying problems, under only Assumption A.4 in Chapter 3 or B.4 (the average distance between two consecutive optima is proportional to the sampling time elapsed) in Chapter 4 without any further stringent state evolution assumption. Moreover, the probabilistic arguments in Chapter 3 and Chapter 4 are non-asymptotic. Additionally, a data-dependent gain-tuning strategy is proposed in Chapter 5.

Some possible future work includes:

- It appears unlikely that the bounds in Chapter 3 that use Lipschitz constants and strong convexity parameters can be improved much. But how to efficiently estimate these needed parameters in an online fashion remains unresolved.

- Most existing works focus on the case where θ_k^* is a singleton for each k for unconstrained optimization. The extension to constrained optimization and multiple minimizers scenarios will help the practical implementation.

  In the constrained or nonsmooth context, the optimum point θ_k^* may not lie within the interior of the feasible region, implying that the gradient at θ_k^* may not be zero. Namely, $\boldsymbol{g}_k(\theta_k^*) = \mathbf{0}$ is no longer a necessary and sufficient condition for determining θ_k^*, and other optimality condition should be discussed.

- It would be of interest to extend the discussion of Theorem 4.1.1 to more involved scenarios such as correlated noise, multi-scale, state-dependent noise processes, decentralized/asynchronous algorithms, and discontinuities in the algorithms.

- Future work on computable probabilistic bound as in Theorem 4.2.1 may consider the extension of the bound to the case where FDSA or SPSA (instead of SGD) is used in time-varying problems (e.g., [108]). The main benefit is that only noisy measurements of the loss function $f_k(\cdot)$ are needed, but the main theoretical complication introduced by FDSA or SPSA is that the gradient estimate is biased.

- Even though Chapter 5 discusses a data-dependent gain-tuning, more theoretical and practical work is still needed to effectively tuning the constant gain to regulate the tracking capability and stability needs. Some unresolved questions relative to gain tuning are listed below.

- The critical value for the change detection in Section 5.1 is data-dependent, which forces us to estimate unknown covariance matrix $\boldsymbol{\Sigma}$ in (5.3) on the fly. If some distribution-free test statistic with high power can be adapted to meet the change detection purpose, that may help streamline the change detection procedure.
- Assumption C.3 (the optimum remains constant within each regime), in some real-world applications, may still be restrictive. The extension to the scenario C.4 (the optimum remains stationary within each regime) will be very much desirable, yet it requires more in-depth understanding of the limiting distribution of $\hat{\theta}_k$, which is currently unavailable.
- Both Sect. 5.1 and Sect. 5.2 require that the sequence of loss functions take the quadratic form $(\theta - \theta_k^*)\boldsymbol{H}_k(\theta - \theta_k^*)/2$. Can we extend the form of loss functions $\{f_k(\cdot)\}$ to more general nonlinear form?
- The explicit form of (5.18), which pertains to the variance of the moving average of the inner product of two consecutive noisy gradient estimates, is difficult to derive. Nonetheless, if that is available, it does help to improve Algorithm 3 that adaptively changes the gain based on observed data θ_k^*.
- How can we select an optimal gain while estimating the drift term and the noise level in an online fashion?
- Are there any values of η_+ and η_- (the parameters that govern the increase and the decrease of the gain sequence) that are optimal in a certain statistical sense,

i.e., the resulting estimate $\hat{\theta}_k$ achieves the information-theoretic Cramer-Rao lower bound for SA contexts [35]?

- When $\theta_k^* = \theta^*$ for all k, can the idea of determining whether $\hat{\theta}_k$ reaches proximity to stationarity be formalized in a way such that the resulting iterates in Algorithm 3 converge to θ^* a.s.?
- Can we extend the scalar gain to a matrix gain, without incurring much extra computational cost? (Appendix A or [126] demonstrate a reduction of $O(p)$ for the standard SA setup without time variation.)

- Throughout our discussion, we promote *few* measurements of the loss function or the gradient at each sampling time τ_k: only one or two parallel measurements are allowed. An increased number of design points at each k can likely produce a tighter bound for the tracking error $\|\hat{\theta}_k - \theta_k^*\|$, even though this goes against the general philosophy of SA. Is there a way to measure the efficiency trade-off for increased sampling?

There are many unresolved questions, especially for the field of data-dependent gain tuning. This work is a step towards fully understanding how $\hat{\theta}_k$ generated from general SA schemes with non-decaying gains, tracks the time variation in θ_k^* and how much we can trust $\hat{\theta}_k$ as an estimate of θ_k^*.

Appendix A

Second-Order SA in High-Dim Problems

A.1 Introduction

SA algorithms have been widely applied in minimization problems where the loss functions and/or the gradient are only accessible through noisy evaluations. Among all the SA algorithms, the second-order simultaneous perturbation stochastic approximation (2SPSA) and the second-order stochastic gradient (2SG) are particularly efficient in high-dimensional problems covering both gradient-free and gradient-based scenarios. However, due to the necessary matrix operations, the per-iteration FLOPs of the original 2SPSA/2SG are $O(p^3)$ with p being the dimension of the underlying parameter. Note that the $O(p^3)$ FLOPs are distinct from the classical SPSA-based per-iteration $O(1)$ cost in terms of the number of noisy function evaluations. In [126], we propose a technique to efficiently implement the 2SPSA/2SG algorithms via the symmetric indefinite

matrix factorization such that the per-iteration floating-point operations (FLOPs) are reduced from $O(p^3)$ to $O(p^2)$. The almost sure convergence and rate of convergence for the newly-proposed scheme are naturally inherited from the original 2SPSA/2SG. The numerical improvement manifests its superiority in numerical studies in terms of computational complexity and numerical stability.

A.1.1 Problem Context

SA has been widely applied in minimization and/or root-finding problems, when only noisy loss function and/or gradient evaluations are accessible. Consider minimizing a differentiable loss function $f(\boldsymbol{\theta}) : \mathbb{R}^p \to \mathbb{R}$, where only noisy evaluations of $f(\cdot)$ and/or its gradient $\boldsymbol{g}(\cdot)$ are accessible. The key distinction between SA and classical deterministic optimization is the presence of noise, which is largely inevitable when the function measurements are collected from either physical experiments or computer simulation. Furthermore, the noise term comes into play when the loss function is only evaluated on a small subset of an entire (inaccessible) dataset as in online training methods popular with neural network and machine learning. In the era of big-data, we deal with applications where solutions are data-dependent such that the cost is minimized over a given set of sampled data rather than the entire distribution. Overall, SA algorithms have numerous applications in adaptive control, natural language processing, facial recognition, and collaborative filtering, just to name but a few.

In modern machine learning, there is a growing need for algorithms to handle high-dimensional problems. Particularly for deep learning, the need arises as the number of parameters (including both weights and bias) explodes quickly as the network depth and width increase. First-order methods based on back-propagation are widely applied, yet they suffer from slow convergence rate in later iterations after a sharp decline during the early iterations. Second-order methods are occasionally utilized to speed up convergence in terms of the number of iterations, but, still, at a computational burden of $O(p^3)$ per-iteration FLOPs.

To achieve a faster convergence rate at a reasonable computational cost, we present a second-order SP method that incurs only $O(p^2)$ per-iteration FLOPs in contrast to the standard $O(p^3)$. The idea of SP is an elegant generalization of a finite difference (FD) scheme and can be applied in both first-order and second-order SA algorithms. Our proposed method rests on the factorization of symmetric indefinite matrices.

A.1.2 Relevant Prior Works

The adaptive second-order methods here differ in fundamental ways from stochastic quasi-Newton and other similar methods in the machine learning literature. First, most of the machine learning-based methods are designed for loss functions of the ERF form; namely, for functions represented as summations, where each summand represents the contribution of one data vector. Such a structure, together with an assumption of strong convexity, has been exploited in [51, 75], and others for stronger convergence results.

Second, first- or second-order derivative information is often assumed to be directly available on the summands in the loss function (e.g., [23, 96, 98]). Ref. [95] also assumes direct information on the Hessian is available in a second-order stochastic method, but allows for loss functions more general than the ERF. Ref. [23] applies the BFGS method to SO, but under a nonstandard setup where noisy Hessian information can be gathered. In our work, we assume that only noisy loss function evaluations or noisy gradient information are available. Third, notions of convergence and rates of convergence are in line with those in deterministic optimization when the loss function (the ERF) is composed of a finite (although possibly large) number of summands. For example, rates of convergence are linear or quadratic as a measure of iteration-to-iteration improvement in the ERF. In contrast, we follow the traditional notion of stochastic approximation, including applicability to general noisy loss functions, no availability of direct derivative information, and stochastic notions of convergence and rates of convergence based on sample-points (in almost surely sense) and convergence in distribution.

Among various SA schemes, SP algorithms are particularly efficient compared with FD methods. Under certain regularity conditions, [101] shows that the SPSA algorithm uses only $1/p$ of the required number of loss function observations needed in the FD form to achieve the same level of MSE for the SA iterates. To further explore the potential of SP algorithms, [105] presents the second-order SP-based methods, including the 2SPSA for applications in the gradient-free case and the 2SG for applications in the gradient-based case. Those methods estimate the Hessian matrix to achieve near-optimal or

optimal convergence rates and can be viewed as the stochastic analogs of the deterministic Newton-Raphson algorithm. Ref. [107] incorporates both a feedback process and an optimal weighting mechanism in the averaging of the per-iteration Hessian estimates to improve the accuracy of the cumulative Hessian estimate in enhanced second-order simultaneous perturbation stochastic approximation (E2SPSA) and enhanced second-order stochastic gradient (E2SG). The guidelines for practical implementation details and the choice of gain coefficients are available in [104]. More details on the related methods are discussed in [11, Chaps. 7–8].

A.1.3 Our Contribution

Refs. [105, 107] show that the 2SPSA/2SG methods can achieve near-optimal or optimal convergence rates with a much smaller number (independent of dimension p) of loss or gradient function evaluations relative to other second-order stochastic methods in [36, 94]. However, after obtaining function evaluations, the per-iteration FLOPs to update the estimate are $O(p^3)$, as discussed below. The computational burden becomes more severe as p gets larger. This is usually the case in many modern machine learning applications. Here we propose a scheme to implement 2SPSA/2SG efficiently via the symmetric indefinite factorization, which reduces the per-iteration FLOPs from $O(p^3)$ to $O(p^2)$. We also show that the proposed scheme inherits the almost sure convergence and the rate of convergence from the original 2SPSA/2SG in [105].

The remainder of the chapter is as follows. Section A.2 reviews the original 2SPSA/2SG in [105] along with the computational complexity analysis. Section A.3 discusses the proposed efficient implementation, while Section A.4 covers the almost sure convergence and asymptotic normality. Numerical studies are in Section A.5. Section A.6 concludes with a discussion of some practical issues.

A.2 Review of 2SPSA/2SG

Before proceeding, let us review the original 2SPSA/2SG algorithms and explain their $O(p^3)$ per-iteration FLOPs.

A.2.1 2SPSA/2SG Algorithm

Following the routine SA framework, we find the root(s) of $\boldsymbol{g}(\boldsymbol{\theta}) \equiv \partial f(\boldsymbol{\theta}) / \partial \boldsymbol{\theta}$ to solve the problem of finding $\arg\min f(\boldsymbol{\theta})$.

Our central task is to streamline the computing procedure, so we do not dwell on differentiating the global minimizer(s) from the local ones. Such root-finding formulation is widely used in the neural network training and other machine learning literature. We consider optimization under two different settings:

1. Only noisy measurements of the loss function, denoted by $y(\boldsymbol{\theta})$ as in Section 2.1, are available.

2. Only noisy measurements of the gradient function, denoted by $\boldsymbol{Y}(\theta)$ as in Section 2.1, are available.

The conditions for noise can be found in [105, Assumptions C.0 and C.2], which include various types of noise such as Gaussian, multiplicative and impulsive noise as special cases. The main updating recursion for 2SPSA/2SG in [105] is

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k \overline{\overline{\boldsymbol{H}}}_k^{-1} \boldsymbol{G}_k(\hat{\theta}_k), k = 0, 1, \cdots, \tag{A.1}$$

where $\{a_k\}_{k\geq 0}$ is a positive decaying scalar gain sequence, $\boldsymbol{G}_k(\hat{\theta}_k)$ is the direct noisy observation or the approximation of the gradient information, and $\overline{\overline{\boldsymbol{H}}}_k$ is the approximation of the Hessian information. The true gradient $\boldsymbol{g}(\hat{\theta}_k)$ is estimated by:

$$\boldsymbol{G}_k(\hat{\theta}_k) = \begin{cases} \dfrac{y(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k) - y(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)}{2c_k\boldsymbol{\Delta}_k}, \text{for 2SPSA,} & \text{(A.2)} \\ \boldsymbol{Y}_k(\hat{\theta}_k), \quad \text{for 2SG,} & \text{(A.3)} \end{cases}$$

where $\boldsymbol{\Delta}_k = [\Delta_{k1}, \ldots, \Delta_{kp}]^T$ is a mean-zero p-dimensional stochastic perturbation vector with bounded inverse moments [106, Assumption B.6$''$ on pp. 183], $1/\boldsymbol{\Delta}_k = \boldsymbol{\Delta}_k^{-1} \equiv (\Delta_{k1}^{-1}, \cdots, \Delta_{kp}^{-1})^T$ is a vector of reciprocals of each nonzero components of $\boldsymbol{\Delta}_k$ ($\boldsymbol{\Delta}_k^{-T}$ is the transpose of $\boldsymbol{\Delta}_k^{-1}$), and $\{c_k\}_{k\geq 0}$ is a positive decaying scalar gain sequence satisfying conditions in [106, Sect. 7.3]. A valid choice for c_k is $c_k = 1/(k+1)^{1/6}$. For the Hessian

estimate $\overline{\overline{\boldsymbol{H}}}_k$, [105] proposes:

$$\begin{cases} \overline{\overline{\boldsymbol{H}}}_k = \boldsymbol{f}_k(\overline{\boldsymbol{H}}_k)\,, & \text{(A.4)} \\ \overline{\boldsymbol{H}}_k = (1-w_k)\overline{\boldsymbol{H}}_{k-1} + w_k\hat{\boldsymbol{H}}_k\,, & \text{(A.5)} \\ \hat{\boldsymbol{H}}_k = \dfrac{1}{2}\left[\dfrac{\delta\boldsymbol{G}_k}{2c_k}\boldsymbol{\Delta}_k^{-T} + \left(\dfrac{\delta\boldsymbol{G}_k}{2c_k}\boldsymbol{\Delta}_k^{-T}\right)^T\right]\,, & \text{(A.6)} \\ \delta\boldsymbol{G}_k = \boldsymbol{G}_k^{(1)}(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k) - \boldsymbol{G}_k^{(1)}(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)\,, \end{cases}$$

where $\boldsymbol{m}_k\colon \mathbb{R}^{p\times p} \to \{\text{positive definite } p\times p \text{ matrices}\}$ is a preconditioning step to guarantee the positive-definiteness of $\overline{\overline{\boldsymbol{H}}}_k$, $\{w_k\}_{k\geq 0}$ is a positive decaying scalar weight sequence, and $\boldsymbol{G}_k^{(1)}(\hat{\theta}_k \pm c_k\boldsymbol{\Delta}_k)$ are one-sided gradient estimates calculated by:

$$\boldsymbol{G}_k^{(1)}(\hat{\theta}_k \pm c_k\boldsymbol{\Delta}_k) = \begin{cases} \dfrac{y(\hat{\theta}_k \pm c_k\boldsymbol{\Delta}_k + \tilde{c}_k\tilde{\boldsymbol{\Delta}}_k) - y(\hat{\theta}_k \pm c_k\boldsymbol{\Delta}_k)}{\tilde{c}_k\tilde{\boldsymbol{\Delta}}_k}, & \text{in 2SPSA,} \\ \boldsymbol{Y}_k(\hat{\theta}_k \pm c_k\boldsymbol{\Delta}_k), & \text{in 2SG,} \end{cases}$$

where $\{\tilde{c}_k\}_{k\geq 0}$ is another positive decaying gain sequence, and $\tilde{\boldsymbol{\Delta}}_k = (\tilde{\Delta}_{k1}, \cdots, \tilde{\Delta}_{kp})^T$ is generated independently from $\boldsymbol{\Delta}_k$, but in the same statistical manner as $\boldsymbol{\Delta}_k$. Some valid choices for w_k include $w_k = 1/(k+1)$ and the asymptotically optimal choices in [107, Eq. (4.2) or Eq. (4.3)]. Ref. [105] considers the special case where $w_k = 1/(k+1)$, i.e., $\overline{\boldsymbol{H}}_k$ is a sample average of the $\hat{\boldsymbol{H}}_j$ for $j = 1, \cdots, k$. Later [107] proposes the E2SPSA and E2SG to obtain more accurate Hessian estimates by taking the optimal selection of weights and feedback-based terms in (A.5) into account. While the focus of this paper is the original 2SPSA/2SG in [105], we also discuss the applicability of the ideas to the E2SPSA/E2SG algorithms in [107]. Note that, independent of p, one iteration of 2SPSA/E2SPSA uses four

noisy measurements $y(\cdot)$, and one iteration of 2SG/E2SG uses three noisy measurements $\boldsymbol{Y}(\cdot)$.

A.2.2 Per-Iteration Computational Cost of $O(p^3)$

The per-iteration computational cost of $O(p^3)$ arises from two steps: one is from the preconditioning step in (A.4), i.e., obtaining $\overline{\overline{\boldsymbol{H}}}_k$; the other is from the descent direction step in (A.1), i.e., obtaining $\overline{\overline{\boldsymbol{H}}}_k^{-1}\boldsymbol{G}_k(\hat{\theta}_k)$. We now discuss the per-iteration computational cost of these two steps in more detail.

Preconditioning The preconditioning step in (A.4) is to guarantee the positive-definiteness of the Hessian estimate $\overline{\overline{\boldsymbol{H}}}_k$. This step is necessary because the updating of $\overline{\boldsymbol{H}}_k$ in (A.5) does not necessarily yield a positive-definite matrix (but $\overline{\boldsymbol{H}}_k$ is guaranteed to be symmetric). One straightforward way is to perform the following transformation:

$$\boldsymbol{m}_k(\overline{\boldsymbol{H}}_k) = (\overline{\boldsymbol{H}}_k\overline{\boldsymbol{H}}_k + \delta_k\boldsymbol{I})^{1/2}, \tag{A.7}$$

where $\delta_k > 0$ is a small *decaying* scalar coefficient [105] and superscript "1/2" denotes the symmetric matrix square root. Let $\lambda_i(\cdot)$ denote the ith eigenvalue of the argument. In that $\lambda_i(\boldsymbol{A}+c\boldsymbol{I}) = \lambda_i(\boldsymbol{A})+c$ for any matrix $\boldsymbol{A}$ and constant c [49, Obs. 1.1.7], we see that (A.7) directly modifies the eigenvalues of $\overline{\boldsymbol{H}}_k\overline{\boldsymbol{H}}_k$ such that $\lambda_i(\overline{\boldsymbol{H}}_k\overline{\boldsymbol{H}}_k+\delta_k\boldsymbol{I}) = \lambda_i(\overline{\boldsymbol{H}}_k\overline{\boldsymbol{H}}_k)+\delta_k$ for $i = 1, ..., p$. When $\delta_k > 0$, all the eigenvalues of $\overline{\boldsymbol{H}}_k\overline{\boldsymbol{H}}_k + \delta_k\boldsymbol{I}$ are strictly positive and, therefore, the resulting $\overline{\overline{\boldsymbol{H}}}_k$ is positive definite. However, (A.7) has a computational cost of $O(p^3)$ due to both the matrix multiplication in $\overline{\boldsymbol{H}}_k\overline{\boldsymbol{H}}_k$ and the matrix square root

computing [48]. Another intuitive transformation is

$$\boldsymbol{m}_k(\overline{\boldsymbol{H}}_k) = \overline{\boldsymbol{H}}_k + \delta_k \boldsymbol{I} \tag{A.8}$$

for a positive and sufficiently large δ_k. Again, applying eigen-decomposition on $\overline{\boldsymbol{H}}_k$, we see that $\lambda_i(\overline{\overline{\boldsymbol{H}}}_k) = \lambda_i(\overline{\boldsymbol{H}}_k) + \delta_k$ for $i = 1, \cdots, p$. Take $\lambda_{\min}(\cdot) = \min_{1 \le i \le p} \lambda_i(\cdot)$ for any argument matrix in $\mathbb{R}^{p \times p}$. Any $\delta_k > |\lambda_{\min}(\overline{\boldsymbol{H}}_k)|$ will result in $\lambda_{\min}(\overline{\overline{\boldsymbol{H}}}_k) > 0$, and, therefore, the output $\overline{\overline{\boldsymbol{H}}}_k$ is positive definite. Unfortunately, (A.8) cannot avoid the $O(p^3)$ cost in estimating $\lambda_{\min}(\overline{\boldsymbol{H}}_k)$.

In addition to the $O(p^3)$ cost in (A.7) and (A.8), the Hessian estimate $\overline{\overline{\boldsymbol{H}}}_k$ may be ill-conditioned, leading to slow convergence. Ref. [127] proposes to replace all negative eigenvalues of $\overline{\boldsymbol{H}}_k$ with values proportional to its smallest positive eigenvalue. Such modification is shown to improve the convergence rate for problems with ill-conditioned Hessian and achieve smaller mean square errors for problems with better-conditioned Hessian compared with original 2SPSA [127]. However, those benefits are gained at the price of computing the eigenvalues of $\overline{\boldsymbol{H}}_k$, which still costs $O(p^3)$.

Descent direction Another per-iteration computational cost of $O(p^3)$ originates from the descent direction computing in (A.1), which is typically computed by solving the linear system for $\boldsymbol{d}_k$: $\overline{\overline{\boldsymbol{H}}}_k \boldsymbol{d}_k = \boldsymbol{G}_k(\hat{\theta}_k)$. The estimate is updated recursively as following:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k \boldsymbol{d}_k \,. \tag{A.9}$$

With the matrix left-division, it is possible to efficiently solve for $\boldsymbol{d}_k$. However, the computation costs of typical methods, such as LU decomposition or singular value decomposition, are still dominated by $O(p^3)$.

Table A.1: Expressions for terms in (A.10)–(A.12). See [106, Sect. 7.8.2] for detailed suggestions.

Algorithm	t_k	b_k	$\boldsymbol{u}_k$	$\boldsymbol{v}_k$
2SPSA [105]	$1-w_k$	$w_k\delta y_k/(4c_k\tilde{c}_k)$	$\tilde{\boldsymbol{\Delta}}_k^{-1}$	$\boldsymbol{\Delta}_k^{-1}$
E2SPSA [107]	1	$w_k[\delta y_k/(2c_k\tilde{c}_k)]/2$ $-w_k[\boldsymbol{\Delta}_k^T\overline{\boldsymbol{H}}_{k-1}\tilde{\boldsymbol{\Delta}}_k]/2$	$\tilde{\boldsymbol{\Delta}}_k^{-1}$	
2SG [105]	$1-w_k$	$w_k/(4c_k)$	$\delta\boldsymbol{G}_k$	
E2SG [107]	1	$w_k/2$	$\delta\boldsymbol{G}_k/(2c_k)-\overline{\boldsymbol{H}}_{k-1}\boldsymbol{\Delta}_k$	

To speed up the original 2SPSA/2SG, [90] proposes to rearrange (A.5) and (A.6) into the following two sequential rank-one modifications:

$$\begin{cases} \overline{\boldsymbol{H}}_k = t_k\overline{\boldsymbol{H}}_{k-1} + b_k\tilde{\boldsymbol{u}}_k\tilde{\boldsymbol{u}}_k^T - b_k\tilde{\boldsymbol{v}}_k\tilde{\boldsymbol{v}}_k^T\,, & \text{(A.10)} \\ \tilde{\boldsymbol{u}}_k = \sqrt{\dfrac{\|\boldsymbol{v}_k\|}{2\|\boldsymbol{u}_k\|}}\left(\boldsymbol{u}_k + \dfrac{\|\boldsymbol{u}_k\|}{\|\boldsymbol{v}_k\|}\boldsymbol{v}_k\right)\,, & \text{(A.11)} \\ \tilde{\boldsymbol{v}}_k = \sqrt{\dfrac{\|\boldsymbol{v}_k\|}{2\|\boldsymbol{u}_k\|}}\left(\boldsymbol{u}_k - \dfrac{\|\boldsymbol{u}_k\|}{\|\boldsymbol{v}_k\|}\boldsymbol{v}_k\right)\,, & \text{(A.12)} \end{cases}$$

where the scalar terms t_k and b_k (A.10), and vectors $\boldsymbol{u}_k$ and $\boldsymbol{v}_k$ in (A.11) and (A.12) are listed in Table A.1. Applying the matrix inversion lemma [106, pp. 513], [90] shows that $\overline{\boldsymbol{H}}_k^{-1}$ can be computed from $\overline{\boldsymbol{H}}_{k-1}^{-1}$ with a cost of $O(p^2)$. However, the positive-definiteness

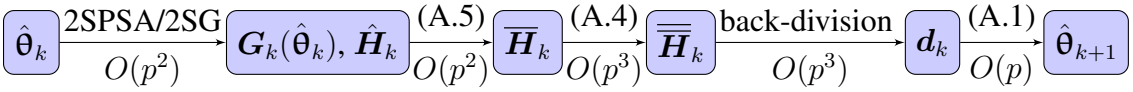
of $\overline{\boldsymbol{H}}_k^{-1}$ is not guaranteed, and an additional eigenvalue modification step similar to either (A.7) or (A.8) is required. As discussed before, for any direct eigenvalue modifications, the computational cost of $O(p^3)$ is inevitable due to the lacking knowledge about the eigenvalues of $\overline{\boldsymbol{H}}_{k-1}^{-1}$.

In short, no prior works can fully streamline the entire second-order SP procedure with an $O\left(p^2\right)$ per-iteration FLOPs, which motivates the elegant procedure below.

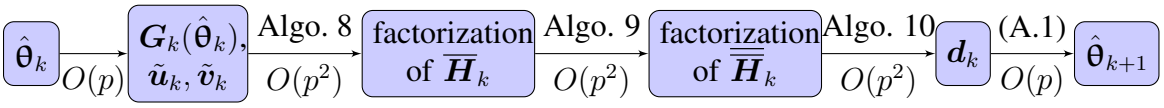
A.3 Efficient Implementation of 2SPSA/2SG

A.3.1 Introduction

With the motivation for proposing an efficient implementation scheme for 2SPSA/2SG laid out in Subsection A.2.2, we now explain our methodology in more detail. Note that none of the prior attempts on 2SPSA/2SG methods can bypass the end-to-end computational cost of $O(p^3)$ per iteration in high-dimensional SO problems. Therefore, we propose replacing $\overline{\boldsymbol{H}}_k$ by its symmetric indefinite factorization, which enables us to implement the 2SPSA/2SG at a per-iteration computational cost of $O(p^2)$. Our work helps alleviate the notorious curse of dimensionality by achieving the fastest possible second-order methods based on Hessian estimation, to the best of our knowledge. Moreover, note that the techniques in [90] are no longer applicable because our scheme keeps track of the matrix factorization instead of the matrix itself, so we propose new algorithms to establish our claims.



(a) Flow chart for the original 2SPSA/2SG



(b) Flow chart for the proposed efficient implementation of 2SPSA/2SG (see Section A.3.4 for detailed description)

Figure A.1: Flow charts showing FLOPs cost at each stage of the original 2SPSA/2SG and the proposed 2SPSA/2SG. Algorithms 8–10 in the lower path are described in Section A.3.3.

To better illustrate our scheme and to be consistent with the original 2SPSA/2SG, we decompose our approach into the following three main steps and discuss the efficient implementation step by step.

i) **Two rank-one modifications**: Update the symmetric indefinite factorization of $\overline{\boldsymbol{H}}_k$ by the two sequential rank-one modifications in (A.10)

ii) **Preconditioning**: Obtain the symmetric indefinite factorization of a positive definite $\overline{\overline{\boldsymbol{H}}}_k$ from the symmetric indefinite factorization of $\overline{\boldsymbol{H}}_k$

iii) **Descent direction**: Update $\hat{\theta}_{k+1}$ by the recursion (A.9)

Note that $\overline{\boldsymbol{H}}_k$ is guaranteed to be symmetric by (A.10) as long as $\overline{\boldsymbol{H}}_0$ is chosen symmetric. For the sake of comparison, we list the flow-charts of the original 2SPSA and that of our proposed scheme in Figure A.1 along with the per-iteration and per-step computational cost. The comparison of the flow-charts helps to put the extra move of indefinite factorization into perspective.

The remainder of this section is as follows. We introduce the symmetric indefinite factorization in Subsection A.3.2 and derive the efficient algorithm in Subsection A.3.3. The per-iteration computational complexity analysis is included in Subsection A.3.4.

A.3.2 Symmetric Indefinite Factorization

This subsection briefly reviews the symmetric indefinite factorization, also called $\boldsymbol{LBL}^T$ factorization, introduced in [20], which applies to any symmetric matrix $\overline{\boldsymbol{H}}$ regardless of the positive-definiteness:

$$\boldsymbol{P}\overline{\boldsymbol{H}}\boldsymbol{P}^T = \boldsymbol{LBL}^T, \tag{A.13}$$

where $\boldsymbol{P}$ is a permutation matrix, $\boldsymbol{B}$ is a block diagonal matrix with diagonal blocks being symmetric with size 1×1 or 2×2, and $\boldsymbol{L}$ is a lower-triangular matrix. Furthermore, the matrices $\boldsymbol{L}$ and $\boldsymbol{B}$ satisfy the following properties [20, Sect. 4], which are fundamental for carrying out subsequent steps i) – iii) at a computational cost of $O\left(p^2\right)$:

- The magnitudes of the entries of $\boldsymbol{L}$ are bounded by a fixed positive constant. Moreover, the diagonal entries of $\boldsymbol{L}$ are all equal to 1.
- $\boldsymbol{B}$ has the same number of positive, negative, and zero eigenvalues as $\overline{\boldsymbol{H}}$.
- The number of negative eigenvalues of $\overline{\boldsymbol{H}}$ is the sum of the number of blocks of size 2×2 on the diagonal and the number of blocks of size 1×1 on the diagonal with

negative entires of $\boldsymbol{B}$. (Note: There are no guarantees for the signs of the entries in the 2×2 blocks.)

The bound on the magnitudes of the entries of $\boldsymbol{L}$ is approximately 2.7808 per [19] and it is *independent* of the size of $\overline{\boldsymbol{H}}$. As shown in Theorem A.4.1–A.4.3, such a constant bound is useful in practice to perform a *quick* sanity check regarding the appropriateness of the symmetric indefinite factorization and to provide useful bounds for the eigenvalues of $\overline{\overline{\boldsymbol{H}}}_k$. From (A.13), $\overline{\boldsymbol{H}}$ can be expressed as $\overline{\boldsymbol{H}} = (\boldsymbol{P}^T\boldsymbol{L})\boldsymbol{B}(\boldsymbol{P}^T\boldsymbol{L})^T$. Then the second bullet point above can be easily shown by *Sylvester's law of inertia*, which states that two congruent matrices have the same number of positive, negative, and zero eigenvalues ($\boldsymbol{A}$ and $\boldsymbol{B}$ are congruent if $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{B}\boldsymbol{P}^T$ for some nonsingular matrix $\boldsymbol{P}$) [109]. From the third bullet point, if $\overline{\boldsymbol{H}}$ is positive semidefinite, the corresponding $\boldsymbol{B}$ is a diagonal matrix with nonnegative diagonal entries.

A.3.3 Algorithm Description

We now illustrate how the $\boldsymbol{L}\boldsymbol{B}\boldsymbol{L}^T$ factorization can be of use in 2SPSA/2SG and discuss steps i) – iii) in Section A.3.1 in detail.

Two rank-one modifications Although the direct calculation of $\overline{\boldsymbol{H}}_k$ in (A.10) only costs $O(p^2)$, the subsequent preconditioning step incurs a computational cost of $O(p^3)$ when not using any factorization of $\overline{\boldsymbol{H}}_k$. Therefore, in anticipation of the subsequent necessary preconditioning, we propose monitoring the $\boldsymbol{L}\boldsymbol{B}\boldsymbol{L}^T$ factorization of $\overline{\boldsymbol{H}}_k$ instead of the matrix itself. That is, the two direct rank-one modifications in (A.10) are

transformed into two non-trivial modifications on the $\boldsymbol{LBL}^T$ factorization, which also incurs a computational cost of $O(p^2)$. It is not necessary that $\overline{\boldsymbol{H}}_k$ is explicitly computed in the algorithm, thereby avoiding the $O(p^3)$ cost arising from matrix-associated necessary multiplications in the preconditioning.

Lemma A.3.1 states that the $\boldsymbol{LBL}^T$ factorization can be updated for rank-one modification at a computational cost of $O(p^2)$. The detailed algorithm is established in [100]. We adopt that algorithm to our two rank-one modifications in (A.10) and present the result in Theorem A.3.1.

Lemma A.3.1. [100, Thm. 2.1]. *Let $\boldsymbol{A} \in \mathbb{R}^{p\times p}$ be symmetric (possibly indefinite) and* non-singular *with $\boldsymbol{PAP}^T = \boldsymbol{LBL}^T$. Suppose that $\boldsymbol{z} \in \mathbb{R}^p$, $\sigma \in \mathbb{R}$ are such that:*

$$\tilde{\boldsymbol{A}} = \boldsymbol{A} + \sigma \boldsymbol{z}\boldsymbol{z}^T \tag{A.14}$$

is also nonsingular. *Then the factorization $\tilde{\boldsymbol{P}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{P}}^T = \tilde{\boldsymbol{L}}\tilde{\boldsymbol{B}}\tilde{\boldsymbol{L}}^T$ can be obtained from the factorization $\boldsymbol{PAP}^T = \boldsymbol{LBL}^T$ with a computational cost of $O(p^2)$.*

Theorem A.3.1. *Suppose $\overline{\boldsymbol{H}}_k$ is given in (A.10). Further, assume that both $\overline{\boldsymbol{H}}_{k-1}$ and $\overline{\boldsymbol{H}}_k$ are* nonsingular *and the factorization $\boldsymbol{P}_{k-1}\overline{\boldsymbol{H}}_{k-1}\boldsymbol{P}_{k-1}^T = \boldsymbol{L}_{k-1}\boldsymbol{B}_{k-1}\boldsymbol{L}_{k-1}^T$ is available. Then the factorization,*

$$\boldsymbol{P}_k\overline{\boldsymbol{H}}_k\boldsymbol{P}_k^T = \boldsymbol{L}_k\boldsymbol{B}_k\boldsymbol{L}_k^T\,, \tag{A.15}$$

can be obtained at a computational cost of $O(p^2)$.

Proof. With Lemma A.3.1, we see that (A.15) can be obtained by applying (A.14) twice with $\sigma = b_k, \boldsymbol{z} = \tilde{\boldsymbol{u}}_k$ and $\sigma = -b_k, \boldsymbol{z} = \tilde{\boldsymbol{v}}_k$, respectively. In as much as each update requires a computational cost of $O(p^2)$, the total computational cost remains $O(p^2)$. □

Remark 18. The nonsingularity (*not* necessarily positive-definiteness) of $\overline{\boldsymbol{H}}_k$ is a modest assumption for the following three reasons: i) $\overline{\boldsymbol{H}}_0$ is often initialized to be a positive definite matrix satisfying the nonsingularity assumption. For example, $\overline{\boldsymbol{H}}_0 = c\boldsymbol{I}$ for some constant $c > 0$. ii) Whenever $\overline{\boldsymbol{H}}_k$ violates the nonsingularity assumption due to the two rank-one modifications in (A.10), a new pair of $\boldsymbol{\Delta}_k$ and $\tilde{\boldsymbol{\Delta}}_k$ along with the noisy measurements can be generated to redo the modifications in (A.10). In practice, the singularity of $\overline{\boldsymbol{H}}_k$ can be detected via the entry-wise bounds of $\boldsymbol{L}_k$ per [19]. Namely, if $\boldsymbol{L}_k$ has an entry exceeding 2.7808, the nonsingularity assumption of $\overline{\boldsymbol{H}}_k$ is violated. It is indeed possible to compute the probability of getting a singular $\overline{\boldsymbol{H}}_k$; however, we deem it as a minor practical issue and do not pursue further analysis in this work. iii) In that the second-order method is often recommended to be implemented only after $\hat{\theta}_k$ reaches the vicinity of θ^*, and the true Hessian matrix of θ^* is assumed to be positive definite [105], the estimate $\overline{\boldsymbol{H}}_k$ is "pushed" towards nonsingularity. The bottom line is that we can run second-order methods at any iteration k, but are more interested when $\hat{\theta}_k$ is near θ^*.

We summarize the two rank-one modifications of $\overline{\boldsymbol{H}}_k$ in Algorithm 8 that follows. The outputs of Algorithm 8 are used to obtain a computational cost of $O(p^2)$ in the preconditioning step as the eigenvalue modifications on $\boldsymbol{B}_k$, a diagonal block matrix, is more efficient than the direct eigenvalue modifications in (A.7) and (A.8). Algorithm 8 is

the key that renders steps ii) and iii) in Subsection A.3.1 achievable at a computational cost of $O(p^2)$.

Algorithm 8 Two rank-one updates of $\overline{\boldsymbol{H}}_k$

Input: matrices $\boldsymbol{P}_{k-1}, \boldsymbol{L}_{k-1}, \boldsymbol{B}_{k-1}$ in the symmetric indefinite factorization of $\overline{\boldsymbol{H}}_{k-1}$, scalars t_k, b_k, and vectors $\boldsymbol{u}_k, \boldsymbol{v}_k$ computed per Table A.1.

Output: matrices $\boldsymbol{P}_k, \boldsymbol{L}_k, \boldsymbol{B}_k$ in the symmetric indefinite factorization of $\overline{\boldsymbol{H}}_k$ per (A.13).

1: **set** $\boldsymbol{P}_k \leftarrow \boldsymbol{P}_{k-1}, \boldsymbol{L}_k \leftarrow \boldsymbol{L}_{k-1}, \boldsymbol{B}_k \leftarrow t_k \boldsymbol{B}_{k-1}$.

2: **update** $\boldsymbol{P}_k, \boldsymbol{L}_k, \boldsymbol{B}_k$ with the rank-one modifications $b_k \tilde{\boldsymbol{u}}_k \tilde{\boldsymbol{u}}_k^T$ with $\tilde{\boldsymbol{u}}_k$ computed in (A.11) and $-b_k \tilde{\boldsymbol{v}}_k \tilde{\boldsymbol{v}}_k^T$ with $\tilde{\boldsymbol{v}}_k$ computed in (A.12), using the updating procedure outlined in [100]. Code is available at `https://github.com/jingyi-zhu/Rank1FactorizationUpdate`.

3: **return** matrices $\boldsymbol{P}_k, \boldsymbol{L}_k, \boldsymbol{B}_k$.

Remark 19. Though $\overline{\boldsymbol{H}}_k$ is not explicitly computed during each iteration, whenever needed, it can be computed easily from its $\boldsymbol{LBL}^T$ factorization, though with a computational cost of $O(p^3)$; i.e, $\overline{\boldsymbol{H}}_k = \boldsymbol{P}_k^T \boldsymbol{L}_k \boldsymbol{B}_k \boldsymbol{L}_k^T \boldsymbol{P}_k$. This calculation yields the same $\overline{\boldsymbol{H}}_k$ as (A.5) or (A.10). The $\boldsymbol{LBL}^T$ factorization of $\overline{\boldsymbol{H}}_0$ requires a computational cost of, at most, $O(p^3)$ [20, Table 2]. However, as a one-time sunk-in cost, it does not compromise the overall computational cost. Of course, we can avoid this bothersome issue by initializing $\overline{\boldsymbol{H}}_0$ to a diagonal matrix, which immediately gives $\boldsymbol{P}_0 = \boldsymbol{L}_0 = \boldsymbol{B}_0 = \boldsymbol{I}$. Generally, the cost for initialization is trivial if $\overline{\boldsymbol{H}}_0$ is a diagonal matrix.

Preconditioning Given the factorization of the estimated Hessian information $\overline{\boldsymbol{H}}_k$, which is symmetric yet potentially *indefinite* (especially during early iterations), we aim to output a factorization of the Hessian approximation $\overline{\overline{\boldsymbol{H}}}_k$ such that $\overline{\overline{\boldsymbol{H}}}_k$ is symmetric and sufficiently positive definite, i.e., $\lambda_{\min}(\overline{\overline{\boldsymbol{H}}}_k) \geq \tau$ for some constant $\tau > 0$. With the above $\boldsymbol{LBL}^T$ factorization associated with $\overline{\boldsymbol{H}}_k$ obtained from the previous two rank-one

modification steps, we can modify the eigenvalues of $\boldsymbol{B}_k$. Note that $\boldsymbol{B}_k$ is a block diagonal matrix, so any eigenvalue modification can be carried out inexpensively. This is in contrast to directly modifying the eigenvalues of $\overline{\boldsymbol{H}}_k$ to obtain $\overline{\overline{\boldsymbol{H}}}_k$, which is computationally-costly as laid out in Subsection A.2.2. Denote $\overline{\boldsymbol{B}}_k$ as the modified matrix from $\boldsymbol{B}_k$. Note that $\overline{\overline{\boldsymbol{H}}}_k$ and $\overline{\boldsymbol{B}}_k$ are congruent as $\overline{\overline{\boldsymbol{H}}}_k = (\boldsymbol{P}_k^T \boldsymbol{L}_k)\overline{\boldsymbol{B}}_k(\boldsymbol{P}_k^T \boldsymbol{L}_k)^T$. By Sylvester's law of inertia, the positive definiteness of $\overline{\overline{\boldsymbol{H}}}_k$ is guaranteed as long as $\overline{\boldsymbol{B}}_k$ is positive definite.

To modify the eigenvalues of $\boldsymbol{B}_k$, we borrow the ideas from the modified Newton's method [80, pp. 50] to set

$$\lambda_j(\overline{\boldsymbol{B}}_k) = \max\{\tau_k, |\lambda_j(\boldsymbol{B}_k)|\}$$

for $j = 1, ..., p$, where τ_k is a user-specified stability threshold, which is possibly data-dependent. A possible choice of the uniformly bounded $\{\tau_k\}$ sequence in the Section A.5 is to set $\tau_k = \max\{10^{-4}, 10^{-4} p \max_{1 \le j \le p} |\lambda_j(\boldsymbol{B}_k)|\}$. The intuition behind the eigenvalue modification in Algorithm 9 is to make $\overline{\boldsymbol{B}}_k$ well-conditioned while behaving similarly to $\boldsymbol{B}_k$. The pseudo-code of the preconditioning step is listed in Algorithm 9.

Algorithm 9 Preconditioning

Input: user-specified stability-threshold $\tau_k > 0$ and matrix $\boldsymbol{B}_k$ in the symmetric indefinite factorization of $\overline{\boldsymbol{H}}_k$.

Output: matrix $\boldsymbol{Q}_k$ in the eigen-decomposition of $\boldsymbol{B}_k$ and the modified matrix $\overline{\boldsymbol{\Lambda}}_k$.

1: **apply** eigen-decomposition of $\boldsymbol{B}_k = \boldsymbol{Q}_k \boldsymbol{\Lambda}_k \boldsymbol{Q}_k^T$, where $\boldsymbol{\Lambda}_k = \text{diag}(\lambda_{k1}, ..., \lambda_{kp})$ and $\lambda_{kj} \equiv \lambda_j(\boldsymbol{B}_k)$ for $j = 1, ..., p$.
2: **update** $\overline{\boldsymbol{\Lambda}}_k = \text{diag}(\bar{\lambda}_{k1}, ..., \bar{\lambda}_{kp})$ with $\bar{\lambda}_{kj} = \max\{\tau_k, |\lambda_{kj}|\}$ for $j = 1, ..., p$.
3: **return** eigen-decomposition of $\overline{\boldsymbol{B}}_k = \boldsymbol{Q}_k \overline{\boldsymbol{\Lambda}}_k \boldsymbol{Q}_k^T$.

Remark 20. Although the eigen-decomposition, in general, incurs an $O(p^3)$ cost, the block diagonal structure of $\boldsymbol{B}_k$ allows such operation to be implemented relatively inexpensively. In the worst-case scenario, $\boldsymbol{B}_k$ consists of $p/2$ diagonal blocks of size 2×2, where eigen-decompositions are applied on each block separately leading to a total computational cost of $O(p)$. For the sake of efficiency, the matrix $\overline{\overline{\boldsymbol{H}}}_k$ is not explicitly computed. Whenever needed, however, it can be computed by $\overline{\overline{\boldsymbol{H}}}_k = \boldsymbol{P}_k^T \boldsymbol{L}_k \boldsymbol{Q}_k \overline{\boldsymbol{\Lambda}}_k \boldsymbol{Q}_k^T \boldsymbol{L}_k^T \boldsymbol{P}_k$ at a cost of $O(p^3)$.

Algorithm 9 makes our approach different from [105]. We only modify the eigenvalues of $\boldsymbol{\Lambda}_k$ (or equivalently of $\boldsymbol{B}_k$), which indirectly affects the eigenvalues of $\overline{\overline{\boldsymbol{H}}}_k$ in a non-trivial way. However, if one constructs $\overline{\boldsymbol{H}}_k$ and $\overline{\overline{\boldsymbol{H}}}_k$ from their factorization (formally unnecessary as mentioned above), Algorithm 9 can be viewed as a function that maps $\overline{\boldsymbol{H}}_k$ to a positive-definite $\overline{\overline{\boldsymbol{H}}}_k$. In this sense, Algorithm 9 is just a special choice of $f_k(\cdot)$ in (A.4) even though such a $f_k(\cdot)$ is non-trivial and difficult to find.

Descent direction After the preconditioning step, the descent direction $\boldsymbol{d}_k$: $\overline{\overline{\boldsymbol{H}}}_k \boldsymbol{d}_k = \boldsymbol{G}_k(\hat{\theta}_k)$ can be computed readily via one forward substitution w.r.t. the lower-triangular matrix $\boldsymbol{L}_k$ and one backward substitution w.r.t. the upper-triangular matrix $\boldsymbol{L}_k^T$, as the decomposition $\overline{\overline{\boldsymbol{H}}}_k = \boldsymbol{P}_k^T \boldsymbol{L}_k \boldsymbol{Q}_k \overline{\boldsymbol{\Lambda}}_k \boldsymbol{Q}_k^T \boldsymbol{L}_k^T \boldsymbol{P}_k$ is available. The estimate $\hat{\theta}_k$ can then be updated as in (A.9). Note that $\overline{\overline{\boldsymbol{H}}}_k$ is not directly computed in any iteration, and the forward and backward substitutions are implemented through the terms in the $\boldsymbol{LBL}^T$ factorization. Algorithm 10 below summarizes the details.

Algorithm 10 Descent Direction Step

Input: gradient estimate $\boldsymbol{G}_k(\hat{\theta}_k)$, and matrices $\boldsymbol{P}_k, \boldsymbol{L}_k, \boldsymbol{Q}_k, \overline{\boldsymbol{\Lambda}}_k$ in the $\boldsymbol{LBL}^T$ factorization of $\overline{\overline{\boldsymbol{H}}}_k$.

Output: descent direction $\boldsymbol{d}_k$.

1: **Solve** $\boldsymbol{z}$ by forward substitution such that $\boldsymbol{L}_k\boldsymbol{z} = \boldsymbol{P}_k\boldsymbol{G}_k(\hat{\theta}_k)$.

2: **Compute** $\boldsymbol{w}$ such that $\boldsymbol{w} = \boldsymbol{Q}_k\overline{\boldsymbol{\Lambda}}_k^{-1}\boldsymbol{Q}_k^T\boldsymbol{z}$.

3: **Solve** $\boldsymbol{y}$ by backward substitution such that $\boldsymbol{L}_k^T\boldsymbol{y} = \boldsymbol{w}$.

4: **return** $\boldsymbol{d}_k = \boldsymbol{P}_k^T\boldsymbol{y}$.

Given the triangular structure of $\boldsymbol{L}_k$ and that both $\boldsymbol{P}_k$ and $\boldsymbol{Q}_k$ are permutation matrices, the computational cost of Algorithm 10 is dominated by $O(p^2)$.

A.3.4 Overall Algorithm (Second-Order SP) and Computational Complexity

With the aforementioned steps, we present the *complete* algorithm for implementing second-order SP in Algorithm 11 below, which applies to 2SPSA/2SG/E2SPSA/E2SG. A complete computational complexity analysis for 2SPSA is also stated, and the suggestions for the user-specified inputs are listed in [106, Sect. 7.8.2]. Results for 2SG/E2SPSA/E2SG can be obtained similarly.

For the terminating condition, the algorithm is set to stop when a pre-specified total number of function evaluations (applicable for 2SPSA and E2SPSA) or gradient measurements (applicable for 2SG and E2SG) is reached or the norm of the differences between several consecutive estimates is less than a pre-specified threshold. Note that, for

Algorithm 11 Efficient Second-order SP (applies to 2SPSA, 2SG, E2SPSA, and E2SG)

Input: initialization $\hat{\theta}_0$ and $\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{B}_0$ in the symmetric indefinite factorization of $\overline{\boldsymbol{H}}_0$; user-specified stability-threshold $\tau_k > 0$; coefficients a_k, c_k, w_k and, for 2SPSA/E2SPSA, $\tilde{c}_k$.

Output: terminal estimate $\hat{\theta}_k$.

1: **set** iteration index $k = 0$.
2: **while** terminating condition for $\hat{\theta}_k$ has not been satisfied **do**
3: **estimate** gradient $\boldsymbol{G}_k(\hat{\theta}_k)$ by (A.2) or (A.3).
4: **compute** $t_k, b_k, \tilde{\boldsymbol{u}}_k$ and $\tilde{\boldsymbol{v}}_k$ by (A.11), (A.12) and Table A.1.
5: **update** the symmetric indefinite factorization of $\overline{\boldsymbol{H}}_k$ by Algorithm 8.
6: **update** the symmetric indefinite factorization of $\overline{\overline{\boldsymbol{H}}}_k$ by Algorithm 9.
7: **compute** the descent direction $\boldsymbol{d}_k$ by Algorithm 10.
8: **update** $\hat{\theta}_{k+1} = \hat{\theta}_k - a_k \boldsymbol{d}_k$.
9: $k \leftarrow k + 1$
10: **end while**
11: **return** $\hat{\theta}_k$.

Table A.2: Computational complexity analysis in gradient-free case (2SPSA in Algorithm 11) Complexity cost shown in FLOPs.

Leading Cost	Original 2SPSA	Proposed Implementation
Update $\overline{\boldsymbol{H}}_k$	$7p^2$	$3.67p^2 + O(p)$
Precondition $\overline{\overline{\boldsymbol{H}}}_k$	$17.67p^3 + O(p^2)$	$8p$
Descent direction $\boldsymbol{d}_k$	$0.33p^3 + O(p^2)$	$4p^2 + O(p)$
Total Cost	$18p^3 + O(p^2)$	$7.67p^2 + O(p)$

each iteration, four noisy loss function measurements are required in the gradient-free case, whereas three noisy gradient measurements are required in the gradient-based case.

The corresponding computational complexity analysis for Algorithm 11 under the gradient-free case is summarized in Table A.2. Analogously, the analysis can be carried out for the gradient-based case and the feedback-based case (E2SPSA or E2SG).

Let us now show how we obtain the terms in Table A.2. A floating-point operation (FLOP) is assumed to be either a summation or a multiplication, while transposition

requires no FLOPs. For the updating $\overline{\boldsymbol{H}}_k$ step in the original 2SPSA, $3p^2$ FLOPs are required per (A.5) and $4p^2$ FLOPs are required per (A.6). In the proposed implementation, $10p$ FLOPs are required to get $\tilde{\boldsymbol{u}}_k$ and $\tilde{\boldsymbol{v}}_k$ per (A.11) and (A.12), respectively, and $22p^2/6 + O(p)$ FLOPs are required to update the symmetric indefinite factorization of $\overline{\boldsymbol{H}}_k$ [100, Thm. 2.1]. For the preconditioning step in the original 2SPSA, if using (A.7), $p^3 + p$ FLOPs are required to get $\overline{\boldsymbol{H}}_k\overline{\boldsymbol{H}}_k + \delta_k\boldsymbol{I}$ and an additional $50p^3/3 + O(p^2)$ FLOPs are required for the matrix square root operation [48]. In the proposed implementation, at most $7p$ FLOPs are required to get an eigenvalue decomposition on $\boldsymbol{B}_k$ (14 FLOPs for at most $p/2$ blocks of size 2×2), and p FLOPs are required to update the eigenvalues of $\boldsymbol{B}_k$. For computing the descent direction $\boldsymbol{d}_k$ in the original 2SPSA, $p^3/3$ FLOPs are required to apply Cholesky decomposition for $\overline{\overline{\boldsymbol{H}}}_k$, and $2p^2$ FLOPs are required for the backward substitutions. In the proposed implementation, $4p^2 + 2p$ FLOPs are required to backward substitutions.

Table A.2 may not provide the lowest possible computational complexities because a great deal of existing work on parallel computing—such as [39] on parallelization of Cholesky decomposition, [24] for computing principal matrix square root, and [30] for the symmetric eigenvalue problem—have tremendously accelerated the matrix-operation computing speed in modern data analysis packages. Nonetheless, even with such enhancements, the FLOPS counts remain $O(p^3)$ in the standard methods. The bottom line is that our proposed implementation reduces the overall computational cost from $O(p^3)$ to $O(p^2)$.

A.4 Theoretical Results and Practical Benefits

This section presents the theoretical foundation related to the almost sure convergence and the asymptotic normality of $\hat{\theta}_k$. We also offer comments on the practical benefits of the proposed scheme. Lemma A.4.1 provides the theoretical guarantee to connect the eigenvalues of $\overline{\overline{\boldsymbol{H}}}_k$ and $\overline{\boldsymbol{\Lambda}}_k$, which are important for proving Theorem A.4.1–A.4.3 related to the matrix properties of $\overline{\boldsymbol{H}}_k$ and $\overline{\overline{\boldsymbol{H}}}_k$.

Lemma A.4.1. [49, Thm. 4.5.9]. *Let $\boldsymbol{A}, \boldsymbol{S} \in \mathbb{R}^{p\times p}$, with $\boldsymbol{A}$ being symmetric and $\boldsymbol{S}$ being* nonsingular. *Let the eigenvalues of $\boldsymbol{A}$ and $\boldsymbol{S}\boldsymbol{A}\boldsymbol{S}^T$ be arranged in* nondecreasing *order. Let $\sigma_1 \geq \cdots \geq \sigma_p > 0$ be the singular values of $\boldsymbol{S}$. For each $j = 1, \cdots, p$, there exists a positive number $\zeta_j \in [\sigma_p^2, \sigma_1^2]$ such that $\lambda_j(\boldsymbol{S}\boldsymbol{A}\boldsymbol{S}^T) = \zeta_j \lambda_j(\boldsymbol{A})$.*

Before presenting the main theorems, we first discuss the singular values of $\boldsymbol{L}_k$. Denote $\{\sigma_i(\boldsymbol{L}_k)\}_{i=1}^p$ as the singular values of $\boldsymbol{L}_k$ and let $\sigma_{\min}(\cdot) = \min_{1\leq i\leq p} \sigma_i(\cdot)$, $\sigma_{\max}(\cdot) = \max_{1\leq i\leq p} \sigma_i(\cdot)$. Since $\boldsymbol{L}_k$ is a unit lower triangular matrix, we have $\lambda_j(\boldsymbol{L}_k) = 1$ for $j = 1, .., p$ and $\det(\boldsymbol{L}_k) = 1$. From the entry-wise bounds of $\boldsymbol{L}_k$ in Subsection A.3.2, we see that $p \leq \|\boldsymbol{L}_k\|_F \leq 3p^2/2 - p/2$ for all k, where $\|\cdot\|_F$ is the Frobenius norm of the argument matrix in $\mathbb{R}^{p\times p}$. With the lower bound of $\sigma_{\min}(\boldsymbol{L}_k)$ [121], there exists a constant $\underline{\sigma} > 0$ such that $\sigma_{\min}(\boldsymbol{L}_k) \geq \underline{\sigma}$ for all k. On the other hand, by the equivalence of the matrix norms, i.e, $\sigma_{\max}(\boldsymbol{L}_k) = \|\boldsymbol{L}_k\|_2 \leq \|\boldsymbol{L}_k\|_F$ for $\|\cdot\|_2$ being the spectral norm, there exists a constant $\overline{\sigma} > 0$ such that $\sigma_{\max}(\boldsymbol{L}_k) \leq \overline{\sigma}$ for all k. Both $\underline{\sigma}$ and $\overline{\sigma}$ are independent of the sample path for $\boldsymbol{L}_k$. By the Rayleigh-Ritz theorem [49, Thm. 4.2.2], $\boldsymbol{e}_1^T(\boldsymbol{L}_k\boldsymbol{L}_k^T)\boldsymbol{e}_1 = 1$

implies that $\sigma_{\min}(\boldsymbol{L}_k) \leq 1$ and $\sigma_{\max}(\boldsymbol{L}_k) \geq 1$. Combined, all the singular values of $\boldsymbol{L}_k$ are bounded uniformly across k; i.e., $\underline{\sigma} < \sigma_{\min}(\boldsymbol{L}_k) \leq 1 \leq \sigma_{\max}(\boldsymbol{L}_k) \leq \overline{\sigma}$. Let $\kappa(\boldsymbol{L}_k)$ be the condition number of $\boldsymbol{L}_k$, then $1 \leq \kappa(\boldsymbol{L}_k) \leq \overline{\sigma}/\underline{\sigma}$.

As the focus of Algorithm 9 is to generate a positive definite $\overline{\boldsymbol{B}}_k$ (or equivalently its eigen-decomposition), we replace τ_k in Theorem A.4.1–A.4.3 with some constant $\underline{\tau} \in (0, \tau_k]$ independent of the sample path for $\boldsymbol{B}_k$ for all k. Note that the substitution is solely for succinctness and does not affect the theoretical result that $\overline{\boldsymbol{B}}_k$ is positive definite. Theorem A.4.1 presents the key theoretical properties of $\overline{\overline{\boldsymbol{H}}}_k$ satisfying the regularity conditions in [105, C.6]. Based on Theorem A.4.1, the strong convergence, $\hat{\theta}_k \to \theta^*$ and $\overline{\boldsymbol{H}}_k \to \boldsymbol{H}(\theta^*)$, can be established conveniently. See Remark 21.

Theorem A.4.1. *Assume there exists a symmetric indefinite factorization* $\overline{\boldsymbol{H}}_k = \boldsymbol{P}_k^T \boldsymbol{L}_k \boldsymbol{B}_k \boldsymbol{L}_k^T \boldsymbol{P}_k$. *Given any constant* $\underline{\tau} \in (0, \tau_k]$ *for all* k, *the matrix* $\overline{\overline{\boldsymbol{H}}}_k = \boldsymbol{P}_k^T \boldsymbol{L}_k \boldsymbol{Q}_k \overline{\boldsymbol{\Lambda}}_k \boldsymbol{Q}_k^T \boldsymbol{L}_k^T \boldsymbol{P}_k$ *with* $\boldsymbol{Q}_k$ *and* $\overline{\boldsymbol{\Lambda}}_k$ *returned from Algorithm 9 satisfies the following properties:*

(a) $\lambda_{\min}(\overline{\overline{\boldsymbol{H}}}_k) \geq \underline{\sigma}^2 \underline{\tau} > 0$.

(b) $\overline{\overline{\boldsymbol{H}}}_k^{-1}$ *exists a.s.,* $c_k^2 \overline{\overline{\boldsymbol{H}}}_k^{-1} \to \boldsymbol{0}$ *a.s., and for some constants* $\delta, \rho > 0$, $\mathbb{E}[\|\overline{\overline{\boldsymbol{H}}}_k^{-1}\|^{2+\delta}] \leq \rho$.

Proof. For all k, it is easy to see that $\lambda_{\min}(\overline{\boldsymbol{\Lambda}}_k) \geq \underline{\tau} > 0$ implying $\overline{\boldsymbol{\Lambda}}_k$ is positive definite. Since both $\boldsymbol{Q}_k$ and $\boldsymbol{L}_k$ are nonsingular, by Sylvester's law of inertia [109], $\overline{\overline{\boldsymbol{H}}}_k$ is also

positive definite as $\overline{\mathbf{\Lambda}}_k$ is positive definite. Moreover, by Lemma A.4.1,

$$\lambda_{\min}(\overline{\overline{\boldsymbol{H}}}_k) \geq \sigma^2_{\min}(\boldsymbol{L}_k)\lambda_{\min}(\overline{\mathbf{\Lambda}}_k) \geq \underline{\sigma}^2\underline{\tau} > 0\,. \tag{A.16}$$

Because $\overline{\overline{\boldsymbol{H}}}_k$ has a constant lower bound for all its eigenvalues across k, property (b) follows. □

Remark 21. Theorem A.4.1 guarantees that $\overline{\overline{\boldsymbol{H}}}_k$ is positive definite, and, therefore, the estimates of θ in the second-order method move in a descent-direction on average. Meeting property (b) is also necessary for showing the convergence results. Suppose the routine regularity conditions in [105, Sect. III and IV] hold. To depict the strong convergence, $\hat{\theta}_k \to \theta^*$ and $\overline{\boldsymbol{H}}_k \to \boldsymbol{H}(\theta^*)$, we need only verify that $\overline{\overline{\boldsymbol{H}}}_k$ satisfies the regularity conditions in [105, C.6] because the key difference between the original 2SPSA/2SG and our proposed method is effectively the preconditioning step. Theorem A.4.1 verifies the Assumption C.6 in [105] directly, and therefore we have $\hat{\theta}_k \to \theta^*$ a.s. and $\overline{\boldsymbol{H}}_k \to \boldsymbol{H}(\theta^*)$ a.s. under both the 2SPSA and 2SG settings by [105, Thms. 1 and 2].

Theorem A.4.2 discusses the connection between $\overline{\boldsymbol{H}}_k$ and $\overline{\overline{\boldsymbol{H}}}_k$ when k is sufficiently large. It also verifies a key condition when proving the asymptotic normality of $\hat{\theta}_k$. See Remark 22.

Theorem A.4.2. *Assume $\boldsymbol{H}(\theta^*)$ is positive definite. When choosing $0 < \underline{\tau} \leq \lambda_{\min}(\boldsymbol{H}(\theta^*))/(2\overline{\sigma}^2)$, there exists a constant K_1 such that for all $k > K_1$, we have $\overline{\overline{\boldsymbol{H}}}_k = \overline{\boldsymbol{H}}_k$.*

Proof. By Remark 21, since $\overline{\boldsymbol{H}}_k \to \boldsymbol{H}(\theta^*)$ a.s., there exists an integer K such that for all $k > K_1$, $\lambda_{\min}(\overline{\boldsymbol{H}}_k) \geq \lambda_{\min}(\boldsymbol{H}(\theta^*))/2 > 0$. By Lemma A.4.1, we can achieve a lower bound for the eigenvalues of $\boldsymbol{\Lambda}_k$ as

$$\lambda_{\min}(\boldsymbol{\Lambda}_k) \geq \frac{\lambda_{\min}(\overline{\boldsymbol{H}}_k)}{\sigma^2_{\max}(\boldsymbol{L}_k)} \geq \frac{\lambda_{\min}(\overline{\boldsymbol{H}}_k)}{\overline{\sigma}^2} \geq \underline{\tau}\,.$$

Therefore, for all $k > K_1$, $\overline{\boldsymbol{\Lambda}}_k = \boldsymbol{\Lambda}_k$ and, consequently, $\overline{\overline{\boldsymbol{H}}}_k = \overline{\boldsymbol{H}}_k$. □

Remark 22. Theorem A.4.2 shows that when k is large (the estimated Hessian $\overline{\boldsymbol{H}}_k$ is sufficiently positive definite), the proposed preconditioning step will automatically make $\overline{\overline{\boldsymbol{H}}}_k = \overline{\boldsymbol{H}}_k$, which satisfies one of the key required conditions for the asymptotic normality of $\hat{\theta}_k$ in [105]. Apart from the additional regularity conditions in [105, C.10–12], we are required to verify that $\overline{\overline{\boldsymbol{H}}}_k - \overline{\boldsymbol{H}}_k \to \mathbf{0}$ a.s., which can be inferred by Theorem A.4.2. Following [105, Thm. 3], when the gain sequences have the standard form $a_k = a/(A + k + 1)^\alpha$ and $c_k = c/(k+1)^\gamma$, the asymptotic normality of $\hat{\theta}_k$ gives:

$$k^{(\alpha-2\gamma)/2}(\hat{\theta}_k - \theta^*) \xrightarrow{\text{dist}} N(\boldsymbol{\mu}, \boldsymbol{\Omega}) \quad \text{for 2SPSA,}$$

$$k^{\alpha/2}(\hat{\theta}_k - \theta^*) \xrightarrow{\text{dist}} N(\mathbf{0}, \boldsymbol{\Omega}') \quad \text{for 2SG,}$$

where the specifications of $\alpha, \gamma, \boldsymbol{\mu}, \boldsymbol{\Omega}$ and $\boldsymbol{\Omega}'$ are available in [105]. Under E2SPSA/E2SG settings, the convergence and asymptotic results can be derived analogously from [107, Thms. 1–4].

As an ill-conditioned matrix may cause an excessive step-size in recursion (A.9) leading to slow a convergence rate [68], we need to make sure that the resulting $\overline{\overline{\boldsymbol{H}}}_k$ (or its equivalent factorization) is not only positive definite but also numerically favorable. Theorem A.4.3 below shows that changing the eigenvalues of $\boldsymbol{\Lambda}_k$ does not lead to the eigenvalues of $\overline{\overline{\boldsymbol{H}}}_k$ becoming either too large or too small.

Theorem A.4.3. *Assume the eigenvalues of* $\boldsymbol{H}(\theta^*)$ *are bounded uniformly such that* $0 < \underline{\lambda}^* < |\lambda_j(\boldsymbol{H}(\theta^*))| < \overline{\lambda}^* < \infty$ *for* $j = 1, ..., p$ *for all* k*. Then there exists some* K_2 *such that for* $k > K_2$*, the eigenvalues and condition number of* $\overline{\overline{\boldsymbol{H}}}_k$ *are also bounded uniformly.*

Proof. Again by Remark 21, in that $\overline{\boldsymbol{H}}_k \to \boldsymbol{H}(\theta^*)$ a.s.; therefore, for all $k > K_2$, the eigenvalues of $\overline{\boldsymbol{H}}_k$ are bounded uniformly in the sense that $\underline{\lambda} < |\lambda_j(\overline{\boldsymbol{H}}_k)| < \overline{\lambda}$ for $j = 1, ..., p$, where $\underline{\lambda} = \underline{\lambda}^*/2$ and $\overline{\lambda} = 2\overline{\lambda}^*$ are constants independent of the sample path for $\overline{\boldsymbol{H}}_k$. Given $\overline{\boldsymbol{H}}_k = \boldsymbol{P}_k \boldsymbol{L}_k \boldsymbol{B}_k \boldsymbol{L}_k^T \boldsymbol{P}_k$, by Lemma A.4.1,

$$\frac{\lambda_{\min}(\overline{\boldsymbol{H}}_k)}{\sigma^2_{\max}(\boldsymbol{L}_k)} \leq \lambda_{\min}(\boldsymbol{B}_k) \leq \frac{\lambda_{\min}(\overline{\boldsymbol{H}}_k)}{\sigma^2_{\min}(\boldsymbol{L}_k)},$$

and

$$\frac{\lambda_{\max}(\overline{\boldsymbol{H}}_k)}{\sigma^2_{\max}(\boldsymbol{L}_k)} \leq \lambda_{\max}(\boldsymbol{B}_k) \leq \frac{\lambda_{\max}(\overline{\boldsymbol{H}}_k)}{\sigma^2_{\min}(\boldsymbol{L}_k)}.$$

Similarly, since $\overline{\overline{\boldsymbol{H}}}_k = \boldsymbol{P}_k \boldsymbol{L}_k \overline{\boldsymbol{B}}_k \boldsymbol{L}_k^T \boldsymbol{P}_k$,

$$\begin{aligned}
\lambda_{\min}(\overline{\overline{\boldsymbol{H}}}_k) &\geq \sigma_{\min}^2(\boldsymbol{L}_k)\lambda_{\min}(\overline{\boldsymbol{B}}_k) \\
&\geq \sigma_{\min}^2(\boldsymbol{L}_k) \max\left\{\underline{\tau}, \frac{\lambda_{\min}(\overline{\boldsymbol{H}}_k)}{\sigma_{\max}^2(\boldsymbol{L}_k)}\right\} \\
&\geq \underline{\sigma}^2 \max\left\{\underline{\tau}, \frac{\underline{\lambda}}{\overline{\sigma}^2}\right\},
\end{aligned}$$

$$\begin{aligned}
\lambda_{\max}(\overline{\overline{\boldsymbol{H}}}_k) &\leq \sigma_{\max}^2(\boldsymbol{L}_k)\lambda_{\max}(\overline{\boldsymbol{B}}_k) \\
&\leq \sigma_{\max}^2(\boldsymbol{L}_k) \max\left\{\underline{\tau}, \frac{\lambda_{\max}(\overline{\boldsymbol{H}}_k)}{\sigma_{\min}^2(\boldsymbol{L}_k)}\right\} \\
&\leq \overline{\sigma}^2 \max\left\{\underline{\tau}, \frac{\overline{\lambda}}{\underline{\sigma}^2}\right\},
\end{aligned}$$

where $\kappa(\cdot)$ is the condition number of the matrix argument. In as much as $\underline{\sigma}^2, \overline{\sigma}^2, \underline{\lambda}$, and $\overline{\lambda}$ are all constants specified before running the algorithm, the eigenvalues of $\overline{\overline{\boldsymbol{H}}}_k$ are bounded uniformly across $k > K_2$.

Moreover, for the condition number of $\overline{\overline{\boldsymbol{H}}}_k$, we have:

$$\kappa(\overline{\overline{\boldsymbol{H}}}_k) \leq \frac{\sigma_{\max}^2(\boldsymbol{L}_k)}{\sigma_{\min}^2(\boldsymbol{L}_k)} \frac{\max\left\{\underline{\tau}, \lambda_{\max}(\overline{\boldsymbol{H}}_k)/\sigma_{\min}^2(\boldsymbol{L}_k)\right\}}{\max\left\{\underline{\tau}, \lambda_{\min}(\overline{\boldsymbol{H}}_k)/\sigma_{\max}^2(\boldsymbol{L}_k)\right\}}.$$

Hence, the condition number of $\overline{\overline{\boldsymbol{H}}}_k$ is also bounded uniformly across $k > K_2$. □

Remark 23. Theorem A.4.3 is highly desired for the preconditioning step as it ensures the numerical stability. Recall that the preconditioning step listed in Algorithm 9 modifies the eigenvalues of $\overline{\boldsymbol{H}}_k$ by modifying the eigenvalues of $\boldsymbol{B}_k$. This modification is desirable because the eigenvalues of $\overline{\overline{\boldsymbol{H}}}_k$ are controllable; i.e., a bound for $\lambda_j(\overline{\overline{\boldsymbol{H}}}_k)$ uniformly for

sufficiently large k under a given size p can be obtained. The controlled condition number in Theorem A.4.3 demarcates the original preconditioning procedure as in Eq. (A.8), which does not control the condition number of $\overline{\overline{\boldsymbol{H}}}_k$.

A.5 Numerical Studies

In this section, we demonstrate the strength of the proposed algorithms by minimizing the skewed-quartic function [105] using efficient 2SPSA/E2SPSA and training a neural network using efficient 2SG.

A.5.1 Skewed-Quartic Function

We consider the following skewed-quartic function used in [105] to show the performance of the efficient 2SPSA/E2SPSA:

$$f(\boldsymbol{\theta}) = \boldsymbol{\theta}^T \boldsymbol{B}^T \boldsymbol{B} \boldsymbol{\theta} + 0.1 \sum_{i=1}^{p} (\boldsymbol{B}\boldsymbol{\theta})_i^3 + 0.01 \sum_{i=1}^{p} (\boldsymbol{B}\boldsymbol{\theta})_i^4 \,,$$

where $(\cdot)_i$ is the ith component of the argument vector, and $\boldsymbol{B}$ is such that $p\boldsymbol{B}$ is an upper triangular matrix of all 1's. The additive noise in $y(\cdot)$ is independent $\mathcal{N}(0, 0.05^2)$; i.e., $y(\boldsymbol{\theta}) = f(\boldsymbol{\theta}) + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, 0.05^2)$. It is easy to check that that $L(\boldsymbol{\theta})$ is strictly convex with a unique minimizer $\boldsymbol{\theta}^* = \mathbf{0}$ such that $L(\boldsymbol{\theta}^*) = 0$.

For the preconditioning step in the original 2SPSA/E2SPSA, we choose $\overline{\overline{\boldsymbol{H}}}_k = \boldsymbol{m}_k(\overline{\boldsymbol{H}}_k) = (\overline{\boldsymbol{H}}_k \overline{\boldsymbol{H}}_k + 10^{-4} e^{-k} \boldsymbol{I})^{1/2}$, which satisfies the definition of $\boldsymbol{m}_k(\cdot)$ in (A.7)

as $\delta_k = 10^{-4}e^{-k} \to 0$. In the efficient 2SPSA/E2SPSA, we choose $\overline{\mathbf{\Lambda}}_k = \text{diag}(\bar{\lambda}_{k1}, ..., \bar{\lambda}_{kp})$ with $\bar{\lambda}_{kj} = \max\{10^{-4}, 10^{-4}p \max_{1\leq i\leq p} |\lambda_{ki}|, |\lambda_{kj}|\}$ for all j, which is consistent with the suggestion in [100, pp. 118] and satisfies Theorem A.4.1. To guard against unstable steps during the iteration process, a blocking step is added to reset $\hat{\boldsymbol{\theta}}_{k+1}$ to $\hat{\boldsymbol{\theta}}_k$ if $\|\hat{\boldsymbol{\theta}}_{k+1} - \hat{\boldsymbol{\theta}}_k\| \geq 1$. We choose an initial value $\hat{\boldsymbol{\theta}}_0 = [1, 1, \ldots, 1]^T$.

We show three plots below. Figures A.2 and A.3 illustrate how the efficient method here provides essentially the same solution in terms of the loss function values as the $O(p^3)$ methods in [105] and [107] (2SPSA and feedback and weighting-based E2SPSA). Figure 4 illustrates how the $O(p^3)$ vs. $O(p^2)$ FLOPS-based cost in Table A.2 above is manifested in overall runtimes.

Figure A.2 plots the normalized loss function values $[f(\hat{\boldsymbol{\theta}}_k) - f(\boldsymbol{\theta}^*)]/[f(\hat{\boldsymbol{\theta}}_0) - f(\boldsymbol{\theta}^*)]$ of the original 2SPSA and the efficient 2SPSA averaged over 20 independent replicates for $p = 100$ and the number of iterations $N = 50,000$. Similar to the numerical studies in [107], the gain sequences of the two algorithms are chosen to be $a_k = a/(A + k + 1)^{0.602}$, $c_k = \tilde{c}_k = c/(k + 1)^{0.101}$, and $w_k = w/(k + 1)^{0.501}$, where $a = 0.04, A = 1000, c = 0.05$, and $w = 0.01$ following the standard guidelines in [104].

Figure A.3 compares the normalized loss function values $[f(\hat{\boldsymbol{\theta}}_k) - f(\boldsymbol{\theta}^*)]/[f(\hat{\boldsymbol{\theta}}_0) - f(\boldsymbol{\theta}^*)]$ of the standard E2SPSA and the efficient E2SPSA averaged over 10 independent replicates for $p = 10$ and number of iterations $N = 10,000$. The gain sequences of the two algorithms are chosen to have the form $a_k = a/(A + k + 1)^{0.602}$, $c_k = \tilde{c}_k = c/(k + 1)^{0.101}$,

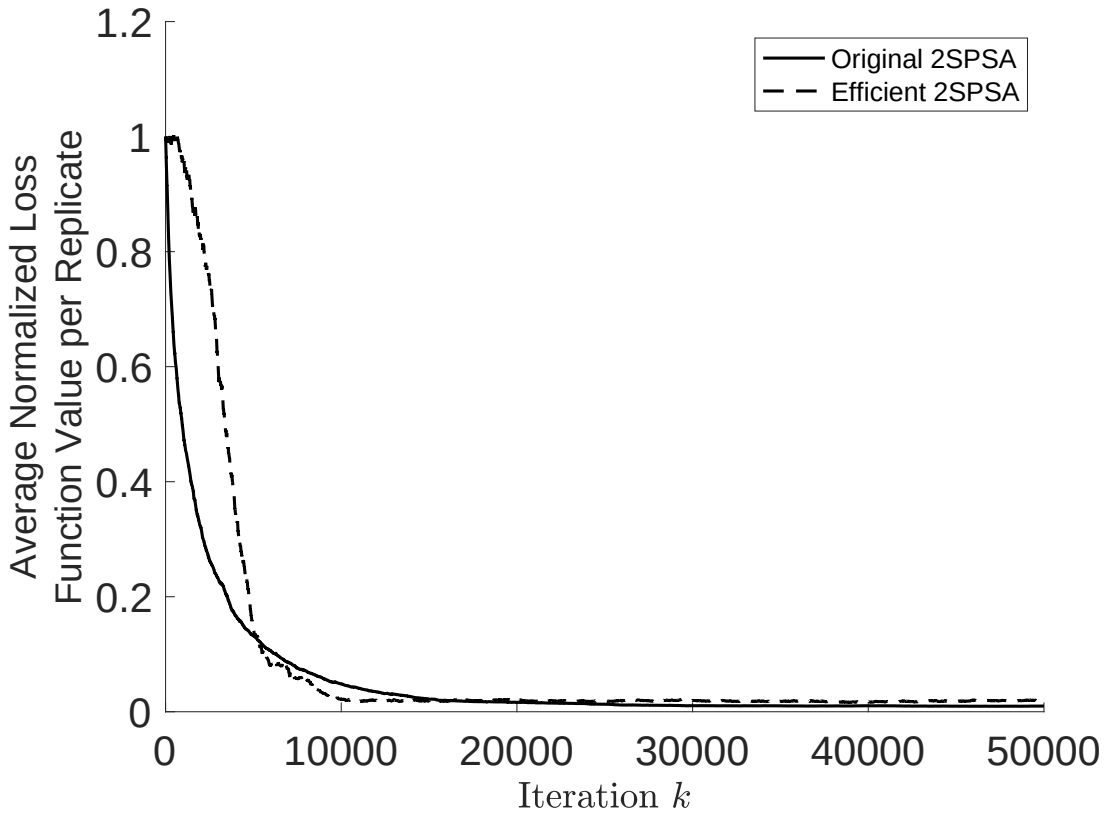


Figure A.2: Similar performance of algorithms with respect to loss values (*different* run times). Normalized terminal loss $[f(\hat{\theta}_k) - f(\theta^*)]/[f(\hat{\theta}_0) - f(\theta^*)]$ of the original 2SPSA and the efficient 2SPSA averaged over 20 replicates for $p = 100$.

and $w_k = w/(k+1)^{0.501}$, where $a = 0.3, A = 50$, and $c = 0.05$. The weight sequence $w_k = \tilde{c}_k^2 c_k^2 / [\sum_{i=0}^{k} (\tilde{c}_i^2 c_i^2)]$ is set according to the optimal weight in [107, Eq. (4.2)].

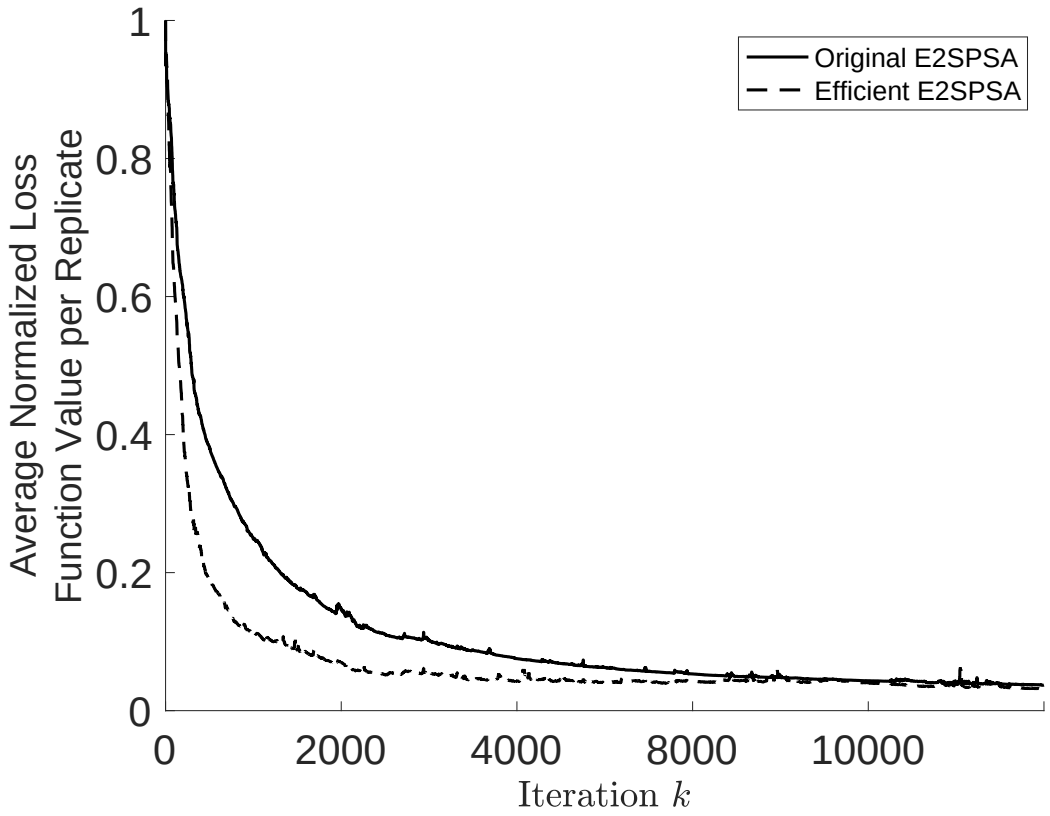


Figure A.3: Similar performance of algorithms with respect to loss values (*different* run times). Normalized terminal loss $[f(\hat{\theta}_k) - f(\theta^*)]/[f(\hat{\theta}_0) - f(\theta^*)]$ of the original E2SPSA and the efficient E2SPSA averaged over 10 replicates for $p = 10$.

In the above comparisons, the loss function decreases significantly for all the dimensions with only noisy loss function measurements available. We see that the two implementations of E2SPSA provide close to the same accuracy for 1000 or more iterations, although at a computing cost difference of $O(p^2)$ versus $O(p^3)$. Note that the differences (across k) between the original 2SPSA and the efficient 2SPSA/E2SPSA in

Figure A.3 can be made arbitrarily small by picking an appropriate $\boldsymbol{m}_k(\cdot)$ (or equivalently $\overline{\overline{\boldsymbol{H}}}_k$) in the original 2SPSA, although such a choice might be non-trivial.

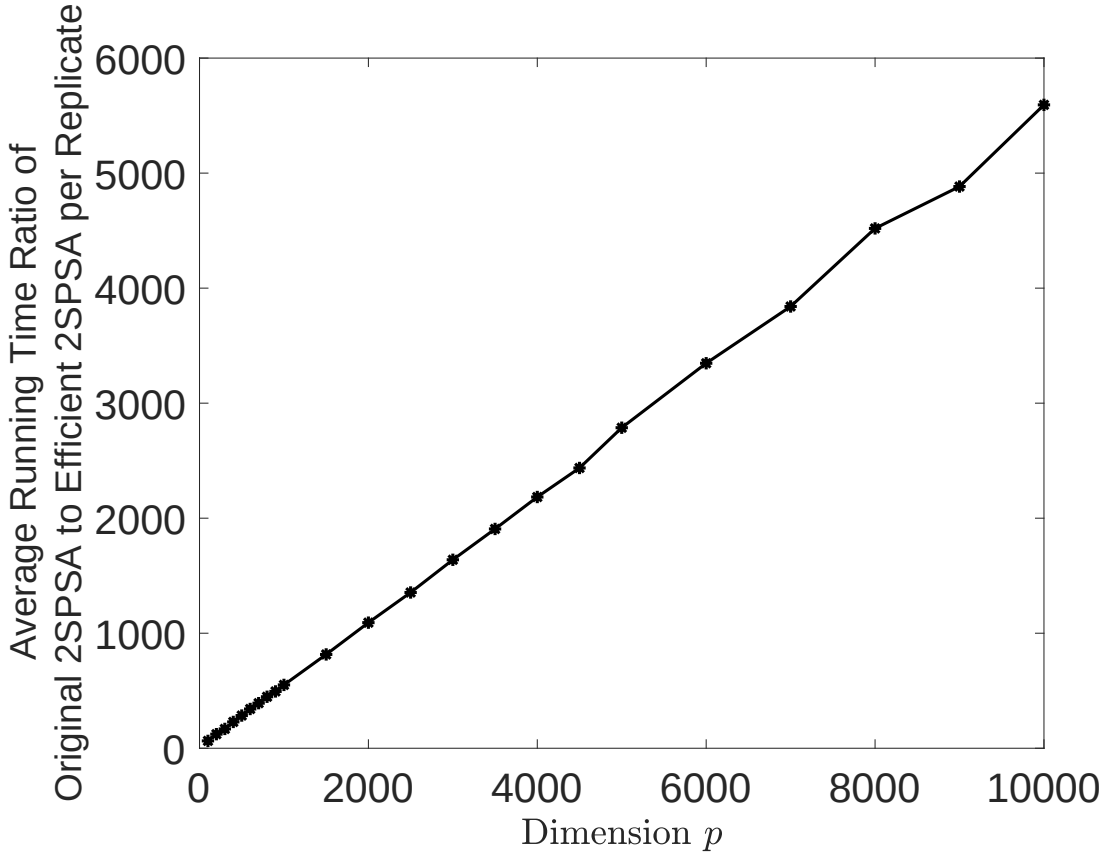


Figure A.4: Running time ratio of the original 2SPSA to the efficient 2SPSA averaged over 10 replicates, where the same skewed-quartic loss function is used, and the total number of iterations is fixed at 10 for each run. The trend is close to the theoretical linear relationship as a function of dimension p.

To measure the computational time, Figure A.4 plots the running time (measured by the built-in C++ function `clock()` with no input) ratio of the original 2SPSA to the efficient 2SPSA averaged over 10 independent replicates with dimension up to 10000. It visualizes the practicality of the efficient 2SPSA over the original 2SPSA. In terms of the general trend, the linear relationship between the running time ratio and the dimension

number is consistent with the $O(p^3)$ cost for the original 2SPSA and the $O(p^2)$ cost for the efficient 2SPSA. From Figure A.4, it is clear that the computational benefit of the efficient 2SPSA is more apparent as the dimension p increases. The slope in Figure A.4 is roughly 0.56, which is consistent with the theoretical FLOPs ratio of 2.35 in Table A.2, when accounting for differences due to the storage costs and code efficiency. With a more dedicated programming language, it is expected that the running time ratio will be closer to the theoretical FLOPs ratio in Table A.2.

A.5.2 Real-Data Study: Airfoil Self-Noise Data Set

In this subsection, we compare the efficient 2SG with the SGD and ADAM [58] in training a one-hidden-layer feed-forward neural network to predict sound levels over an airfoil. Although there are many gradient-based methods to train a neural network, we select SGD and ADAM because they are popular and representative of algorithms within the machine learning community. Comparison of efficient 2SG and the two aforementioned algorithms is appropriate as all of them use the noisy gradient evaluations *only*, despite their different forms. Aside from the application here, neural networks have been widely used as function approximators in the field of aerodynamics and aeroacoustics. Recent applications include airfoil design [89], and aerodynamic prediction [81].

The dataset used in this example is the NASA data of the NACA 0012 airfoil self-noise data set [17, 18], which is also available on the UC Irvine Machine Learning Repository at `https://archive.ics.uci.edu/ml/datasets/Airfoil+Self-Noise`.

This NASA dataset is obtained from a series of aerodynamic and acoustic tests of two and three-dimensional airfoil blade sections conducted in an anechoic wind tunnel. The inputs contain five variables: frequency (in Hertz); angle of attack (in degrees, not in radians); chord length (in meters); free-stream velocity (in meters per second); and suction side displacement thickness (in meters). The output contains the scaled sound pressure level (in decibels). Readers may refer to [18] and [32, Sect. 3] for further details.

Now that the number of samples is $n = 1503$, we fit the dataset using a one-hidden-layer neural network with 150 hidden neurons and with sigmoid activating functions. Other choices in neural network structures, that use a different number of layers or different activation functions, have been implemented in [32]. Here, we use a neural network with a greater number of neurons than in [32] to demonstrate the strength of the efficient 2SG in high-dimensional problems. The value of p is 1051, calculated as 5×150 weights and 150 bias parameters for the hidden neurons along with 150 weights and 1 bias parameters for the output neuron.

Following the principles in [118], we train the neural network in an online manner, where only one training sample is evaluated during each iteration. Denote the dataset as $\{(y_i, \boldsymbol{x}_i)\}_{i=1}^n$ and the parameters in the neural network as $\boldsymbol{\theta}$. The loss function is chosen to be the ERF; i.e., $f(\boldsymbol{\theta}) = (1/n)\sum_{i=1}^n (y_i - \hat{y}_i)^2$, where $\hat{y}_i$ is the neural network output based on input $\boldsymbol{x}_i$ and parameter $\boldsymbol{\theta}$. Consistent with the online training of an ERF in machine learning, the loss function based on that one training sample can be deemed as a noisy measurement of the loss function based on the entire dataset.

We implement SGD and ADAM with 10 epochs, each corresponding to 1503 iterations (one iteration per data point), resulting in a total of 15030 iterations. The gain sequence is chosen to be $a_k = a/(k+1+A)^\alpha$ and $A = 1503$ being 10% of the total number of iterations with $\alpha = 1$ following [106, pp. 113–114]. After tuning for optimal performance, we choose $a = 1$ for SGD and ADAM [58] . Other hyper-parameters for ADAM are determined from the default settings in [58]. There is no "re-setting" of a_k imposed at the beginning of each epoch so that the gain sequence goes down consecutively across iterations and epochs. The initial value $\hat{\theta}_0 = \mathbf{0}$. Recall that efficient 2SG requires three back-propagations per iteration, where SGD and ADAM only require one per iteration. Therefore, for fair comparison, we implement the efficient 2SG under two different scenarios: (1) serial computing, and (2) concurrent computing.

Within each iteration of efficient 2SG, the three gradient measurements, $\boldsymbol{Y}_k(\hat{\theta}_k), \boldsymbol{Y}_k(\hat{\theta}_k + c_k\boldsymbol{\Delta}_k)$ and $\boldsymbol{Y}_k(\hat{\theta}_k - c_k\boldsymbol{\Delta}_k)$ can be computed simultaneously in as much as they do not rely on each other. Using this concurrent implementation, the time spent in back-propagation can be reduced to one-third of the original time. All the remaining steps are unchanged. Although the efficient 2SG takes time in performing Algorithm 9, numerical studies indicate that the majority of the time is spent on back-propagation. Therefore, under the concurrent implementation, the efficient 2SG has roughly the same running time per iteration as SGD and ADAM. Figure A.5 shows the value of ERF under the concurrent implementation. In the efficient 2SG, the gain sequences are chosen to be $a_k = a/(A+k+1)^\alpha$, $w_k = 1/(k+1)$, and $c_k = c/(k+1)^\gamma$

with $A = 1503, \alpha = 1$ and $\gamma = 1/6$ following [104]. Other parameters of $a = 0.1$ and $c = 0.05$ are tuned for optimal performance. The matrix $\overline{\mathbf{\Lambda}}_k$ is computed the same as in the skewed-quartic function above. For better practical performance, training data is normalized to the range $[0, 1]$. As all the inputs and outputs are positive, normalization is simply performed by dividing the data by their corresponding maximums. Figure A.5 shows that the efficient 2SG converges much quicker and obtains a better terminal value. One explanation for this phenomenon is that the Hessian information helps the speed of convergence, similar to the benefits of Newton-Raphson relative to the gradient-descent method.

Figure A.6 compares the ERF of the two algorithms in terms of the number of gradient evaluations. Note that each iteration of SGD and ADAM takes one gradient evaluation, while the efficient 2SG necessitates three. This comparison is suitable for the non-concurrent implementation because one iteration of the efficient 2SG has roughly the cost of three iterations of the SGD. It is shown in Figure A.6 that the efficient 2SG still outperforms SGD and ADAM even without any concurrent implementation. There is less than a 7% difference in running time among SGD, ADAM, and the efficient 2SG under the concurrent implementation.

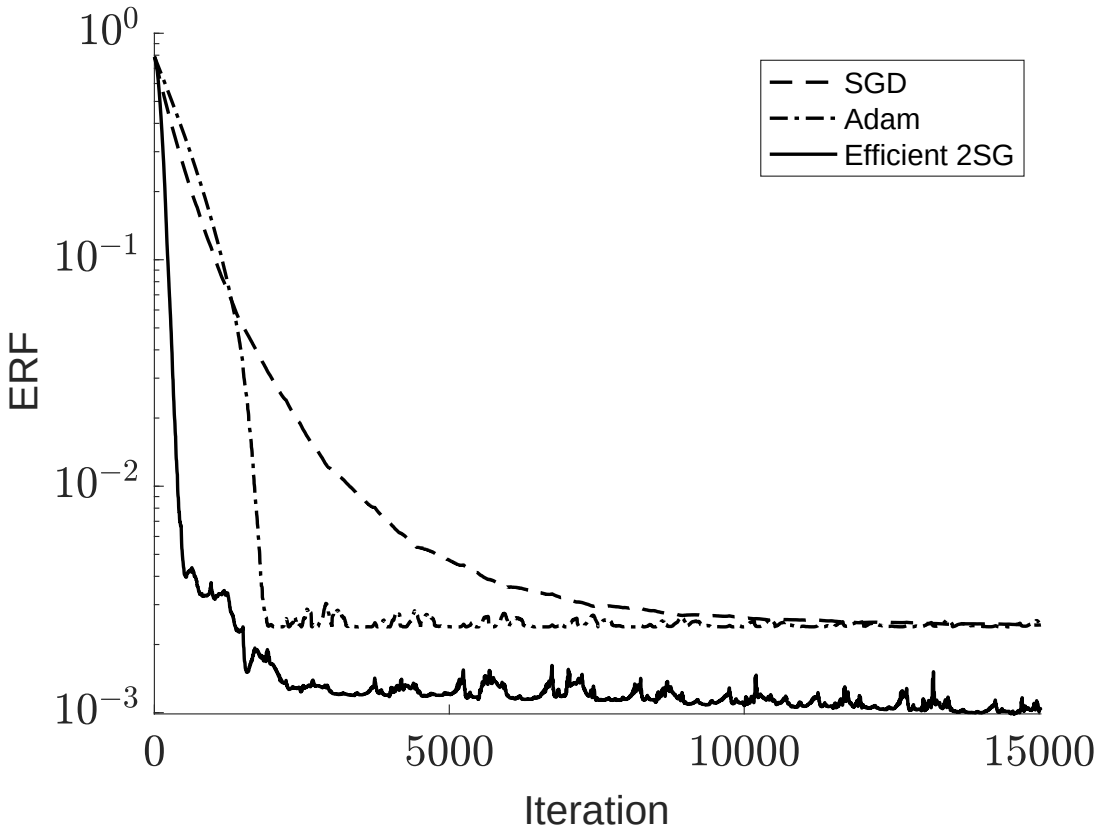


Figure A.5: ERF of training samples in SGD, ADAM, and the efficient 2SG under concurrent implementation.

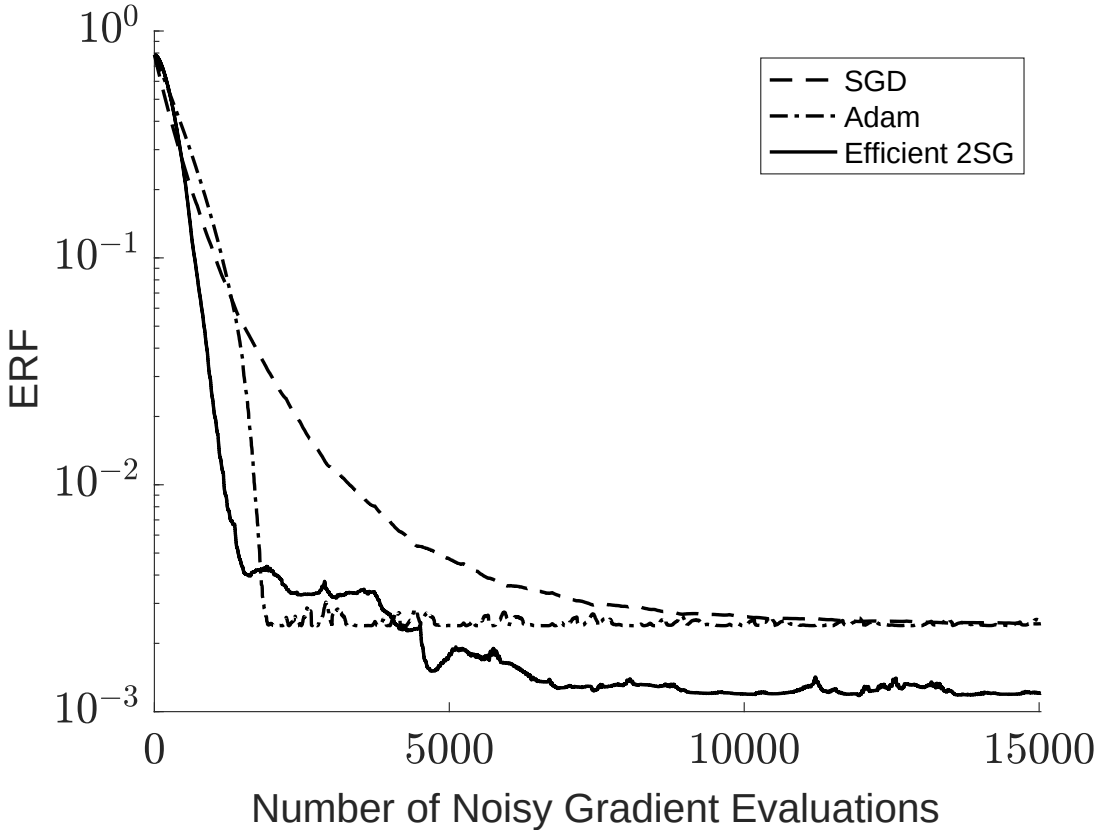


Figure A.6: ERF of training samples in SGD, ADAM, and the efficient 2SG per gradient evaluation under *serial* (non-concurrent) computing. SGD and ADAM have three times the number of iterations of 2SG.

A.6 Practical Issues and Concluding Remarks

Let us discuss two practical questions regarding Algorithm 11. i) What is the difference between the standard adaptive SPSA-based method and the proposed algorithm if $\overline{\boldsymbol{B}}_k$ (or $\overline{\boldsymbol{H}}_k$) is sufficiently positive definite? ii) How to recover $\overline{\boldsymbol{H}}_k$ at any k?

In the ideal case, if $\boldsymbol{B}_k$ (or $\overline{\boldsymbol{H}}_k$) is assumed to always be positive definite, the preconditioning step becomes unnecessary and we can directly set the symmetric indefinite factorization of $\overline{\boldsymbol{H}}_k$ as the symmetric indefinite factorization of $\overline{\overline{\boldsymbol{H}}}_k$; i.e., $\boldsymbol{\Lambda}_k = \overline{\boldsymbol{\Lambda}}_k$. In this scenario, the proposed method is identical to the original 2SPSA. However, because of the symmetric indefinite factorization, the overall computational cost remains at $O(p^2)$ as in Table A.2, and it is still favorable relative to the original 2SPSA, which incurs a computational cost of $O(p^3)$ due to the Gaussian elimination of $\overline{\overline{\boldsymbol{H}}}_k$ in computing the descent direction $\boldsymbol{d}_k$. As mentioned in Section A.2.2, however, [90] uses the matrix inversion lemma to show that the computational cost can be reduced to $O(p^2)$ as well. Compared with [90], which directly updates the matrix $\overline{\boldsymbol{H}}_k^{-1}$ using the matrix inverse lemma, our proposed method has more control over the eigenvalues of $\overline{\boldsymbol{H}}_k$ and performs well even when $\overline{\boldsymbol{H}}_k$ is ill-conditioned.

The second aspect is that both $\overline{\boldsymbol{H}}_k$ and $\overline{\overline{\boldsymbol{H}}}_k$ are never explicitly computed during each iteration. By maintaining the corresponding factorization, we avoid expensive matrix multiplications and gain a much faster way to achieve second-order convergence. However, whenever needed, either $\overline{\boldsymbol{H}}_k$ or $\overline{\overline{\boldsymbol{H}}}_k$ can be directly computed from the factorizations at a cost of $O(p^3)$. See Subsection A.3.3.

APPENDIX A. SECOND-ORDER SA IN HIGH-DIM PROBLEMS

To the best of our knowledge, 2SPSA, 2SG, E2SPSA, and E2SG are the fastest possible second-order stochastic Newton-type algorithms based on the estimations of the Hessian matrix from either noisy loss measurements or noisy gradient measurements. This paper shows how symmetric indefinite matrix factorization may be used to reduce the per-iteration FLOPs of the algorithms from $O(p^3)$ to $O(p^2)$. The approach guarantees both a positive definite estimation of the Hessian matrix ("preconditioned") as well as a valid stochastic Newton-type update of the parameter vector, both in $O(p^2)$. This implementation scheme serves to improve practical performance in high-dimensional problems, such as deep learning. In our proposed scheme, formal convergence and convergence rates for $\hat{\theta}_k$ and $\overline{\boldsymbol{H}}_k$ are maintained, following the prior work [105, 107].

Apart from the theoretical guarantee, numerical studies show that the efficient implementation of second-order SP methods provides a promising convergence rate at a tolerable computing cost compared with the stochastic gradient descent method. Note that second-order methods do not provide global convergence in general, and, therefore, the second-order method is recommended to be implemented after reaching the vicinity of the optimizer.

Overall, our proposed scheme of second-order SA methods has values in high-dimensional optimization and learning problems. In that a key step of this work is the symmetric indefinite factorization, the proposed algorithm might be useful for other algorithms whenever updating an estimated Hessian matrix is involved, such as second-order random directions stochastic approximation [87], natural gradient descent

[2], and stochastic variants of the BFGS quasi-Newton methods [96]. In all these methods, instead of directly updating the matrix of interest (usually the Hessian matrix), one might consider updating its corresponding symmetric indefinite factorization in the manner of this paper in order to speed up any matrix inverse operation or matrix eigenvalue modification. Overall, the proposed approach provides a practical second-order method that can be used following first-order or other methods that can place the iterate in at least the vicinity of the solution.

Bibliography

[1] V. M. Alekseev. An estimate for the perturbations of the solutions of ordinary differential equations. *Westnik Moskov Unn. Ser*, 1:28–36, 1961.

[2] S.-I. Amari, H. Park, and K. Fukumizu. Adaptive method of realizing natural gradient learning for multilayer perceptrons. *Neural Computation*, 12(6):1399–1409, 2000.

[3] B. Bamieh and L. Giarre. Identification of linear parameter varying models. *International Journal of Robust and Nonlinear Control*, 12(9):841–853, 2002.

[4] M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes: Theory and Application*, volume 104. Prentice Hall Englewood Cliffs, 1993.

[5] R. E. Bellman. *Stability Theory of Differential Equations*. McGraw-Hill, 1953.

[6] A. Benveniste, M. Métivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximations*, volume 22. Springer Science & Business Media, 2012.

BIBLIOGRAPHY

[7] A. Benveniste and G. Ruget. A measure of the tracking capability of recursive stochastic algorithms with constant gains. *IEEE Transactions on Automatic Control*, 27(3):639–649, 1982.

[8] D. P. Bertsekas. *Convex Optimization Theory*. Athena Scientific Belmont, 2009.

[9] O. Besbes, Y. Gur, and A. Zeevi. Non-stationary stochastic optimization. *Operations Research*, 63(5):1227–1244, 2015.

[10] B. Bharath and V. S. Borkar. Stochastic approximation algorithms: Overview and recent trends. *Sadhana*, 24(4–5):425–452, 1999.

[11] S. Bhatnagar, H. L. Prasad, and L. A. Prashanth. *Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods*, volume 434. Springer, 2013.

[12] P. J. Bickel and K. A. Doksum. *Mathematical Statistics: Basic Ideas And Selected Topics*, volume 1. CRC Press, 2007.

[13] P. Billingsley. *Convergence of Probability Measures*. Wiley, 1968.

[14] A. Blakney and J. Zhu. A comparison of the finite difference and simultaneous perturbation gradient estimation methods with noisy function evaluations. In *Proceedings of the 53rd Conference on Information Science and Systems*, Baltimore, MD, 20–22 March 2019. IEEE.

[15] J. R. Blum. Approximation methods which converge with probability one. *The Annals of Mathematical Statistics*, 25(2):382–386, 1954.

[16] F. Brauer. Perturbations of nonlinear systems of differential equations. *Journal of Mathematical Analysis and Applications*, 14(2):198–206, 1966.

[17] T. F. Brooks and T. H. Hodgson. Trailing edge noise prediction from measured surface pressures. *Journal of Sound and Vibration*, 78(1):69–117, 1981.

[18] T. F. Brooks, D. S. Pope, and M. A. Marcolini. Airfoil self-noise and prediction. Technical Report NASA-RP-1218, L-16528, NAS 1.61:1218, NASA Langley Research Center, Hampton, VA, 1989.

[19] J. R. Bunch and L. Kaufman. Some stable methods for calculating inertia and solving symmetric linear systems. *Mathematics of Computation*, 31(137):163–179, 1977.

[20] J. R. Bunch and B. N. Parlett. Direct methods for solving symmetric indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 8(4):639–655, 1971.

[21] P. A. Burrough, R. McDonnell, R. A. McDonnell, and C. D. Lloyd. *Principles of Geographical Information Systems*. Oxford university press, 2015.

[22] J. C. Butcher. *Numerical Methods for Ordinary Differential Equations*. John Wiley & Sons, 2016.

[23] R. H. Byrd, S. L. Hansen, J. Nocedal, and Y. Singer. A stochastic quasi-Newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.

[24] E. Deadman, N. J. Higham, and R. Ralha. Blocked Schur algorithms for computing the matrix square root. In *Proceedings of the International Workshop on Applied Parallel Computing*, pages 171–182, Helsinki, Finland, 10–13 June 2012.

[25] A. Défossez and F. R. Bach. Averaged least-mean-squares: Bias-variance trade-offs and optimal sampling distributions. In *18th International Conference on Artificial Intelligence and Statistics*, volume 38. Journal of Machine Learning Research, 2015.

[26] B. Delyon and A. Juditsky. Asymptotical study of parameter tracking algorithms. *SIAM Journal on Control and Optimization*, 33(1):323–345, 1995.

[27] D. Derevitskii and A. Fradkov. Two models for analysing the dynamics of adaptation algorithms. *Automation and Remote Control*, 35(1):59–67, 1974.

[28] A. Dieuleveut, A. Durmus, and F. Bach. Bridging the gap between constant step size stochastic gradient descent and markov chains. Available at `https://arxiv.org/abs/1707.06386`, 2017.

[29] P. S. Diniz. *Adaptive Filtering: Algorithms and Practical Implementations*. Springer, 2008.

[30] J. J. Dongarra and D. C. Sorensen. A fully parallel algorithm for the symmetric eigenvalue problem. *SIAM Journal on Scientific and Statistical Computing*, 8(2):139–154, 1987.

[31] Y. Ermoliev. On the method of generalized stochastic gradients and quasi-fejer sequences. *Cybernetics*, 5:208–220, 1969.

[32] L. Errasquin. *Airfoil self-noise prediction using neural networks for wind turbines*. PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 2009.

[33] S. N. Ethier and T. G. Kurtz. *Markov Processes: Characterization and Convergence*, volume 282. John Wiley & Sons, 2005.

[34] E. Eweda and O. Macchi. Tracking error bounds of adaptive nonstationary filtering. *Automatica*, 21(3):293–302, 1985.

[35] V. Fabian. On asymptotic normality in stochastic approximation. *The Annals of Mathematical Statistics*, 39(4):1327–1332, 1968.

[36] V. Fabian. Stochastic approximation. In *Optimizing Methods in Statistics*, pages 439–470. Elsevier, 1971.

[37] D. Farden. Tracking properties of adaptive signal processing algorithms. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(3):439–446, 1981.

[38] J. H. Friedman and L. C. Rafsky. Multivariate generalizations of the Wald-Wolfowitz and Smirnov two-sample tests. *The Annals of Statistics*, pages 697–717, 1979.

[39] A. George, M. T. Heath, and J. Liu. Parallel Cholesky factorization on a shared-memory multiprocessor. *Linear Algebra and Its Applications*, 77:165–187, 1986.

[40] U. Grenander and M. I. Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(4):549–581, 1994.

[41] J. Guckenheimer and P. Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*. Springer-Verlag, Berlin and New York, 1983.

[42] S. Gunnarsson and L. Ljung. Frequency domain tracking characteristics of adaptive algorithms. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(7):1072–1089, 1989.

[43] L. Guo. Estimating time-varying parameters by the Kalman filter based algorithm: stability and convergence. *IEEE Transactions on Automatic Control*, 35(2):141–147, 1990.

[44] L. Guo and L. Ljung. Performance analysis of general tracking algorithms. *IEEE Transactions on Automatic Control*, 40(8):1388–1402, 1995.

[45] F. Gustafsson. *Adaptive Filtering and Change Detection*, volume 1. Wiley, 2000.

[46] P. L. Hall and D. K. Ross. Incoherent neutron scattering functions for random jump diffusion in bounded and infinite media. *Molecular Physics*, 42(3):673–682, 1981.

[47] E. Hazan, A. Rakhlin, and P. L. Bartlett. Adaptive online gradient descent. In *Advances in Neural Information Processing Systems*, pages 65–72, 2008.

[48] N. J. Higham. Computing real square roots of a real matrix. *Linear Algebra and Its Applications*, 88:405–430, 1987.

[49] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

[50] S. Johansen. The Welch-James approximation to the distribution of the residual sum of squares in a weighted linear regression. *Biometrika*, 67(1):85–92, 1980.

[51] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

[52] J. A. Joslin and A. J. Heunis. Law of the iterated logarithm for a constant-gain linear stochastic gradient algorithm. *SIAM Journal on Control and Optimization*, 39(2):533–570, 2000.

[53] H. Kesten. Accelerated stochastic approximation. *The Annals of Mathematical Statistics*, 29(1):41–59, 1958.

[54] H. K. Khalil. *Noninear Systems*. Prentice-Hall, New Jersey, 3rd edition, 2002.

[55] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.

[56] D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 180–191. VLDB Endowment, 2004.

[57] W. Kim, K. Mechitov, J. Y. Choi, and S. Ham. On target tracking with binary proximity sensors. In *Proceedings of the International Symposium on Information Processing in Sensor Netowrks*, pages 301–308, 2005.

[58] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *The International Conference on Learning Representations*, 2015.

[59] K. Krishnamoorthy and J. Yu. Modified Nel and Van der Merwe test for the multivariate Behrens–Fisher problem. *Statistics & Probability Letters*, 66(2):161–169, 2004.

[60] T. G. Kurtz. *Approximation of Population Processes*, volume 36. SIAM, 1981.

[61] H. J. Kushner. *Approximation and Weak Convergence Methods for Random Processes with Applications to Stochastic Systems Theory*. MIT press, 1984.

[62] H. J. Kushner and D. S. Clark. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, volume 26. Springer Science & Business Media, 1978.

[63] H. J. Kushner and H. Huang. Asymptotic properties of stochastic approximations with constant coefficients. *SIAM Journal on Control and Optimization*, 19(1):87–105, 1981.

[64] H. J. Kushner and J. Yang. Analysis of adaptive step-size SA algorithms for parameter tracking. *IEEE Transactions on Automatic Control*, 40(8):1403–1410, 1995.

[65] H. J. Kushner and G. G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer Science & Business Media, 2003.

[66] S. G. Lee, Y. Diaz-Mercado, and M. Egerstedt. Multirobot control using time-varying density functions. *IEEE Transactions on Robotics*, 31(2):489–493, 2015.

[67] Y. Lepage. A combination of Wilcoxon's and Ansari-Bradley's statistics. *Biometrika*, 58(1):213–217, 1971.

[68] X.-L. Li. Preconditioned stochastic gradient descent. *IEEE Transactions on Neural Networks and Learning Systems*, 29(5):1454–1466, 2018.

[69] L. Ljung. Analysis of recursive stochastic algorithms. *IEEE Transactions on Automatic Control*, 22(4):551–575, 1977.

[70] L. Ljung. On positive real transfer functions and the convergence of some recursive schemes. *IEEE Transactions on Automatic Control*, 22(4):539–551, 1977.

[71] L. Ljung and S. Gunnarsson. Adaptation and tracking in system identification—a survey. *Automatica*, 26(1):7–21, 1990.

[72] L. Ljung and P. Priouret. A result on the mean square error obtained using general tracking algorithms. *International Journal of Adaptive Control and Signal Processing*, 5(4):231–248, 1991.

[73] L. Ljung and T. Söderström. *Theory and Practice of Recursive Identification*. MIT press, 1983.

[74] O. Macchi. Optimization of adaptive identification for time-varying filters. *IEEE Transactions on Automatic Control*, 31(3):283–287, 1986.

[75] J. Martens and R. Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417, 2015.

[76] J. L. Maryak, J. C. Spall, and G. L. Silberman. Uncertainties for recursive estimators in nonlinear state-space models, with applications to epidemiology. *Automatica*, 31(12):1889–1892, 1995.

[77] R. C. Merton. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3(1–2):125–144, 1976.

[78] D. Nel and C. Van der Merwe. A solution to the multivariate Behrens-Fisher problem. *Communications in Statistics—Theory and Methods*, 15(12):3719–3735, 1986.

[79] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

[80] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Science & Business Media, 2006.

[81] P. Pérez, A. Trier, and J. Reyes. Prediction of PM2.5 concentrations several hours in advance using neural networks in Santiago, Chile. *Atmospheric Environment*, 34(8):1189–1196, 2000.

[82] C. K. Peterson, A. J. Newman, and J. C. Spall. Simulation-based examination of the limits of performance for decentralized multi-agent surveillance and tracking of undersea targets. In *Signal Processing, Sensor/Information Fusion, and Target Recognition XXIII*, volume 9091, page 90910F. International Society for Optics and Photonics, 2014.

[83] G. C. Pflug. Stochastic minimization with constant step-size: asymptotic laws. *SIAM Journal on Control and Optimization*, 24(4):655–666, 1986.

[84] G. C. Pflug. Stepsize rules, stopping times and their implementation in stochastic quasi-gradient algorithms. *Numerical Techniques for Stochastic Optimization*, pages 353–372, 1988.

[85] B. T. Polyak. *Introduction to Optimization*. Optimization Software, Publications Division, New York, 1987.

[86] A. Y. Popkov. Gradient methods for nonstationary unconstrained optimization problems. *Automation and Remote Control*, 66(6):883–891, 2005.

[87] L. A. Prashanth, S. Bhatnagar, M. Fu, and S. Marcus. Adaptive system optimization using random directions stochastic approximation. *IEEE Transactions on Automatic Control*, 62(5):2223–2238, 2017.

[88] Y. V. Prokhorov. Convergence of random processes and limit theorems in probability theory. *Theory of Probability & Its Applications*, 1(2):157–214, 1956.

[89] M. M. Rai and N. K. Madavan. Aerodynamic design using neural networks. *American Institute of Aeronautics and Astronautics Journal*, 38(1):173–182, 2000.

[90] P. Rastogi, J. Zhu, and J. C. Spall. Efficient implementation of enhanced adaptive simultaneous perturbation algorithms. In *Proceedings of the 50th Conference on Information Science and Systems*, pages 298–303, Princeton, NJ, 16–18 March 2016. IEEE.

[91] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[92] G. J. Ross, D. K. Tasoulis, and N. M. Adams. Nonparametric monitoring of data streams for changes in location and scale. *Technometrics*, 53(4):379–389, 2011.

[93] W. Rudin. *Principles of Mathematical Analysis*. McGraw-Hill Publishing Co., 1976.

[94] D. Ruppert. A Newton-Raphson version of the multivariate Robbins-Monro procedure. *The Annals of Statistics*, 13(1):236–245, 1985.

[95] S. S. Saab and D. Shen. Multidimensional gains for stochastic approximation. *IEEE Transactions on Neural Networks and Learning Systems,* in press at `http://dx.doi.org/10.1109/TNNLS.2019.2920930`, 2019.

[96] N. N. Schraudolph, J. Yu, and S. Günter. A stochastic quasi-Newton method for online convex optimization. In *Artificial Intelligence and Statistics*, pages 436–443, 2007.

[97] A. Simonetto. Time-varying convex optimization via time-varying averaged operators. Available at `https://arxiv.org/abs/1704.07338`, 2017.

[98] J. Sohl-Dickstein, B. Poole, and S. Ganguli. Fast large-scale optimization by unifying stochastic gradient and quasi-Newton methods. In *International Conference on Machine Learning*, pages 604–612, 2014.

[99] V. Solo and X. Kong. *Adaptive Signal Processing Algorithms: Stability and Performance*. Prentice-Hall, Inc., 1994.

[100] D. C. Sorensen. Updating the symmetric indefinite factorization with applications in a modified Newton's method. Technical Report ANL-77-49, Argonne National Laboratory, Argonne, IL, 1977.

[101] J. C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992.

[102] J. C. Spall. Developments in stochastic optimization algorithms with gradient approximations based on function measurements. In *Proceedings of Winter Simulation Conference*, pages 207–214, Orlando, FL, 1994. IEEE.

[103] J. C. Spall. A one-measurement form of simultaneous perturbation stochastic approximation. *Automatica*, 33(1):109–112, 1997.

[104] J. C. Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 34(3):817–823, 1998.

[105] J. C. Spall. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Transactions on Automatic Control*, 45(10):1839–1853, 2000.

[106] J. C. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. John Wiley & Sons, 2003.

[107] J. C. Spall. Feedback and weighting mechanisms for improving Jacobian estimates in the adaptive simultaneous perturbation algorithm. *IEEE Transactions on Automatic Control*, 54(6):1216–1229, 2009.

[108] J. C. Spall and J. A. Cristion. Model-free control of nonlinear stochastic systems with discrete-time measurements. *IEEE Transactions on Automatic Control*, 43(9):1198–1210, 1998.

[109] J. J. Sylvester. A demonstration of the theorem that every homogeneous quadratic polynomial is reducible by real orthogonal substitutions to the form of a sum of positive and negative squares. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 4(23):138–142, 1852.

[110] G. Teschl. *Ordinary Differential Equations and Dynamical Systems*, volume 140. American Mathematical Society Providence, 2012.

[111] J. A. Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8(1–2):1–230, 2015.

[112] H. Walk. An invariance principle for the Robbins-Monro process in a hilbert space. *Probability Theory and Related Fields*, 39(2):135–150, 1977.

[113] L. Wang, J. Zhu, and J. C. Spall. Mixed simultaneous perturbation stochastic approximation for gradient-free optimization with noisy measurements. In *Proceedings of the 59th American Control Conference*, pages 3774–3779, Milwaukee, WI, 27–29 June 2018. IEEE.

[114] Q. Wang and M. Ye. Rate of convergence analysis of simultaneous perturbation stochastic approximation algorithm for time-varying loss function. In *American Control Conference*, pages 5192–5197. IEEE, 2014.

[115] B. Widrow, J. McCool, M. G. Larimore, and C. R. Johnson. Stationary and nonstationary learning characteristics of the LMS adaptive filter. In *Aspects of Signal Processing*, pages 355–393. Springer, 1977.

[116] S. Wiggins. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*, volume 2. Springer Science & Business Media, 2003.

[117] C. Wilson, V. V. Veeravalli, and A. Nedić. Adaptive sequential stochastic optimization. *IEEE Transactions on Automatic Control*, 64(2):496–509, 2018.

[118] D. R. Wilson and T. R. Martinez. The general inefficiency of batch training for gradient descent learning. *Neural Networks*, 16(10):1429–1451, 2003.

[119] Y. Yao. An approximate degrees of freedom solution to the multivariate Behrens Fisher problem. *Biometrika*, 52(1/2):139–147, 1965.

[120] F. Yousefian, A. Nedić, and U. V. Shanbhag. On stochastic gradient and subgradient methods with adaptive steplength sequences. *Automatica*, 48(1):56–67, 2012.

[121] Y. Yu and D. Gu. A note on a lower bound for the smallest singular value. *Linear Algebra and Its Applications*, 253(1–3):25–38, 1997.

[122] J. Zhu and J. C. Spall. Error bound analysis of the least-mean-squares algorithm in linear models. In *Proceedings of the 49th Conference on Information Science and Systems*, Baltimore, MD, 18–20 March 2015. IEEE.

[123] J. Zhu and J. C. Spall. Tracking capability of stochastic gradient algorithm with constant gain. In *Proceedings of the 55th Conference on Decision and Control*, pages 4522–4527, Las Vegas, NV, 12–14 December 2016. IEEE.

[124] J. Zhu and J. C. Spall. Probabilistic bounds in tracking a discrete-time varying process. In *Proceedings of the 57th Conference on Decision and Control*, pages 4849–4854, Miami Beach, FL, 11–13 December 2018. IEEE.

[125] J. Zhu and J. C. Spall. Stochastic approximation with non-decaying gain: Error bounds and data-driven gain-tuning. *International Journal of Robust and Nonlinear Control,* under revision, 2020.

[126] J. Zhu, L. Wang, and J. C. Spall. Efficient implementation of second-order stochastic approximation algorithms in high-dimensional problems. *IEEE Transactions on Neural Networks and Learning Systems,* in press at `http://dx.doi.org/10.1109/TNNLS.2019.2935455`, 2019.

[127] X. Zhu and J. C. Spall. A modified second-order SPSA optimization algorithm for finite samples. *International Journal of Adaptive Control and Signal Processing*, 16(5):397–409, 2002.

Curriculum Vitae

Jingyi Zhu was born in November 1991 in Heyuan City, Guangdong Province, People's Republic of China. Jingyi is currently a Ph.D. candidate in the Department of Applied Mathematics and Statistics (AMS) at the Johns Hopkins University (JHU). Her doctoral study was supported in part by the Paul V. Renoff Fellowship from JHU, the Charles and Catherine Counselman Fellowship from the Department of AMS, the Acheson J. Duncan Fund for the Advancement of Research in Statistics from the Department of AMS, and Navy contract N00024-13-D6400 via the Office of Naval Research. Her efforts in teaching were recognized through the Professor Joel Dean Award for Excellence in Teaching in 2017. In summer 2018, she conducted research on mixed-variable constrained optimization at the JHU Applied Physics Laboratory. In winter 2017, she was an Academic Cooperation Program intern in the Lawrence Livermore National Lab, funded by the National Science Foundation Mathematical Sciences Graduate Internship Program. She received an M.S.E. degree in Computer Science and an M.S.E. degree in Financial Mathematics from JHU, in May 2019 and December 2014, respectively. She received an B.S. degree in Mathematics and Applied Mathematics from Tongji University, Shanghai, China in

CURRICULUM VITAE

May 2013. Broadly, her research lies in the intersection of stochastic optimization and control—particularly the theoretical foundation for stochastic optimization algorithms to be applied in nonstationarity tracking. She also works on general stochastic approximation [14], second-order methods [126], and mixed-variable optimization [113]. In her leisure time, she plays badminton, participates in aerial dancing and yoga, and cooks Cantonese and Hakka cuisine.