Sampling Controlled Stochastic Recursions: Applications to Simulation Optimization and Stochastic Root Finding.

Fatemeh Sadat Hashemi

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Industrial and Systems Engineering

Raghu Pasupathy, Co-Chair
Michael R. Taaffe, Co-Chair
Ebru K. Bish
Scotland C. Leman

September 21, 2015
Blacksburg, Virginia

Keywords: simulation optimization, sampling controlled stochastic recursions, adaptive learning, Brain-Computer Interface

Sampling Controlled Stochastic Recursions: Applications to Simulation Optimization and Stochastic Root Finding.

Fatemeh Sadat Hashemi

(ABSTRACT)

We consider unconstrained Simulation Optimization (SO) problems, that is, optimization problems where the underlying objective function is unknown but can be estimated at any chosen point by repeatedly executing a Monte Carlo (stochastic) simulation. SO, introduced more than six decades ago through the seminal work of Robbins and Monro (and later by Kiefer and Wolfowitz), has recently generated much attention. Such interest is primarily because of SO's flexibility, allowing the implicit specification of functions within the optimization problem, thereby providing the ability to embed virtually any level of complexity. The result of such versatility has been evident in SO's ready adoption in fields as varied as finance, logistics, healthcare, and telecommunication systems.

While SO has become popular over the years, Robbins and Monro's original stochastic approximation algorithm and its numerous modern incarnations have seen only mixed success in solving SO problems. The primary reason for this is stochastic approximation's explicit reliance on a sequence of algorithmic parameters to guarantee convergence. The theory for choosing such parameters is now well-established, but most such theory focuses on asymptotic performance. Automatically choosing parameters to ensure good finite-time performance has remained vexingly elusive, as evidenced by continuing efforts six decades after the introduction of stochastic approximation! The other popular paradigm to solve SO is what has been called sample-average approximation. Sample-average approximation, more a philosophy than an algorithm to solve SO, attempts to leverage advances in modern nonlinear programming by first constructing a deterministic approximation of the SO problem using a fixed sample size, and then applying an appropriate nonlinear programming method. Sample-average approximation is reasonable as a solution paradigm but again suffers from finite-time inefficiency because of the simplistic manner in which sample sizes are prescribed. It turns out that in many SO contexts, the effort expended to execute the Monte Carlo oracle is the single most computationally expensive operation. Sample-average approximation essentially ignores this issue since, irrespective of where in the search space an incumbent solution resides, prescriptions for sample sizes within sample-average approximation remain the same. Like stochastic approximation, notwithstanding beautiful asymptotic theory, sample-average approximation suffers from the lack of automatic implementations that guarantee good finite-time performance.

In this dissertation, we ask: can advances in algorithmic nonlinear programming

theory be combined with intelligent sampling to create solution paradigms for SO that perform well in finite-time while exhibiting asymptotically optimal convergence rates? We propose and study a general solution paradigm called Sampling Controlled Stochastic Recursion (SCSR). Two simple ideas are central to SCSR: (i) use any recursion, particularly one that you would use (e.g., Newton and quasi-Newton, fixed-point, trust-region, and derivative-free recursions) if the functions involved in the problem were known through a deterministic oracle; and (ii) estimate objects appearing within the recursions (e.g., function derivatives) using Monte Carlo sampling "to the extent required." The idea in (i) exploits advances in algorithmic nonlinear programming. The idea in (ii), with the objective of ensuring good finite-time performance and optimal asymptotic rates, minimizes Monte Carlo sampling by attempting to balance the estimated proximity of an incumbent solution with the sampling error stemming from Monte Carlo. This dissertation studies the theoretical and practical underpinnings of SCSR, leading to implementable algorithms to solve SO. We first analyze SCSR in a general context, identifying various sufficient conditions that ensure convergence of SCSR's iterates to a solution. We then analyze the nature of such convergence. For instance, we demonstrate that in SCSRs which guarantee optimal convergence rates, the speed of the underlying (deterministic) recursion and the extent of Monte Carlo sampling are intimately linked, with faster recursions permitting a wider range of Monte Carlo effort. With the objective of translating such asymptotic results into usable algorithms, we formulate a family of SCSRs called Adaptive SCSR (A-SCSR) that adaptively determines how much to sample as a recursion evolves through the search space. A-SCSRs are dynamic algorithms that identify sample sizes to balance estimated squared bias and variance of an incumbent solution. This makes the sample size (at every iteration of A-SCSR) a stopping time, thereby substantially complicating the analysis of the behavior of A-SCSR's iterates. That A-SCSR works well in practice is not surprising — the use of an appropriate recursion and the careful sample size choice ensures this. Remarkably, however, we show that A-SCSRs are convergent to a solution and exhibit asymptotically optimal convergence rates under conditions that are no less general than what has been established for stochastic approximation algorithms.

We end with the application of a certain A-SCSR to a parameter estimation problem arising in the context of brain-computer interfaces (BCI). Specifically, we formulate and reduce the problem of probabilistically deciphering the electroencephalograph (EEG) signals recorded from the brain of a paralyzed patient attempting to perform one of a specified set of tasks. Monte Carlo simulation in this context takes a more general view, as the act of drawing an observation from a large dataset accumulated from the recorded EEG signals. We apply A-SCSR to nine such datasets, showing that in most cases A-SCSR achieves correct prediction

rates that are between 5 and 15 percent better than competing algorithms. More importantly, due to the incorporated adaptive sampling strategies, A-SCSR tends to exhibit dramatically better efficiency rates for comparable prediction accuracies.

Dedication

To my parents.

Acknowledgments

First and foremost, I would like to express my deepest gratitude and acknowledgment to Dr. Raghu Pasupathy, my advisor, for his insightful advice, encouragement, and compassionate support throughout my PhD career. I sincerely appreciate his insight, guidance, kindness, and support. He encouraged me to grow as an independent thinker, and to develop my own individuality and self-sufficiency. For everything you have done for me, Raghu, I thank you.

I would also like to extend my appreciation to Dr. Soumyadip Ghosh; his mentorship at IBM, was paramount in providing a well rounded experience consistent my long-term career goals. I am very grateful for all his wonderful contribution to this research, and the time that he worked closely with us and puzzled over many of the same problems.

I would like to gratefully thanks Dr. Michael R. Taaffe, for his unending encouragement and support, valuable discussions and accessibility. Many thanks to Dr. Scotland Leman; I enjoyed taking Bayesian Statistics class with him, in which I was urged to apply my theoretical research on a real-world application problem, and pushed to get started with an applied thinking. I would like to sincerely thank Dr. Ebru Bish for generously sharing her time and ideas, and also the wonderful opportunity she gave me to present this research in Healthcare Operations Research class, getting wonderful insight and helpful feedback. Most importantly thanks a lot for her friendship during my graduate studies at Virginia Tech.

Additionally, I am so grateful for having such wonderful friends in Blacksburg. They have always been supportive and made it possible for me to live away from my family. Among those amazing friends, I want to specifically thank Sahar Sadeghi, Dr. Nima Mahmoodi, Susan Bixler, Dr. Masoud Agah, and Dr. Leyla Nazhandali for their support and continuous encouragement throughout this journey.

Last but not least, I would like to express my eternal gratitude and appreciation and love to my parents and my wonderful husband, Hodjat, for their unconditional support and never-ending sacrifices.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

We consider unconstrained Simulation Optimization (SO) problems [47], that is, optimization problems where the underlying objective function is unknown but can be estimated at any chosen point by repeatedly executing a Monte Carlo (stochastic) simulation. Formally, the SO problem variation we consider is stated as

$$\text{Problem } P: \quad \begin{array}{ll} \text{minimize} & f(x) \\ x \in \mathbb{X}, & \end{array} \tag{1.1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a twice differentiable function that is bounded from below. For each $x \in \mathbb{X} \in \mathbb{R}^d$, $f(x)$ is estimated using the "consistent estimator" $F(m, x)$ satisfying $F(m, x) \overset{\text{wp1}}{\to} f(x)$ as $m \to \infty$, where m is general and might represent the number of simulation replications (sample size) in the case of terminating simulations or the simulation run length in the case of non-terminating simulations [58]. For instance, the estimator $F(m, x)$ is often the simple sample mean $m^{-1} \sum_{j=1}^{m} Y_j(x)$, where $Y_j(x), j = 1, 2, \ldots, m$ are m independent and identically distributed (iid) replicates obtained by "executing" a Monte Carlo simulation at the point x. Also, depending on the context at hand, unbiased estimates [6] of the gradient function $\nabla f(x)$ may be available along with the function observations used to construct the estimator $F(m, x)$. This is often the case in parameter estimation problems that are posed as Problem P, one of which we discuss in detail in Chapter 5. In such settings, Problem P reduces to that of finding the (vector) zero of a function that can only be obsserved using Monte Carlo estimates. The latter variation of Problem P has been called the Stochastic Root Finding Problem (SRFP) [84, 81, 86] and is the subject of much recent study.

SO problems and SRFPs have recently generated great attention owing to their flexible problem statements. Unlike more rigid variations that insist on specifying the analytic form of the underlying objective function, SO problems as specified above allow implicit representation through a stochastic simulation. This has facilitated adoption across a wide variety of application contexts having virtually any level of "real-world" complexity. Some traditional examples include logistics [48, 50, 8], healthcare [1, 32, 29], and vehicular-traffic systems [64]. More recently, with the advent of big-data contexts, SO and SRFPs have found astonishing relevance when "simulation" (as a verb) is viewed slightly more generally. A large database of either off-line or streaming data from an appropriate context (e.g., natural language processing, electro-encephalograph readings, meteorological data, stock-market ticker data) is accumulated in part with the intent of constructing a predictive model of a desired but unknown quantity. The predictive model is usually a parametric probability model whose parameters are to be determined using the data at hand. Such an estimation problem is then cast as Problem P (with x representing the unknown parameters) by taking the broad but useful view that drawing an observation from the accumulated database is "simulating." Since databases in such contexts tend to be enormous, the traditional way of using every observation in the database towards estimation becomes infeasible, particularly when parameter estimates and corresponding predictions are to be updated rapidly. The question of how to estimate parameters efficiently, that is, using draws from the database parsimoniously, becomes very relevant. Chapter 5 details one such problem from the electro-encephalograph context. For additional examples of SO including downloadable simulation oracles, see `www.simopt.org` [83]. Considering current relevance, an entire track has been devoted to SO and its applications in recent years of the Winter Simulation Conference (WSC).

1.1 Sample Average Approximation and Stochastic Approximation

Two key approaches have emerged in the literature to solve SO problems: Sample Average Approximation (SAA), and Stochastic Approximation (SA). SAA, more a philosophy than an algorithm to solve SO problems, follows a simply stated idea. Instead of solving Problem P, solve a "sample-path" Problem P_{m^*} (to optimality) to obtain a solution estimator X_{m^*}. Formally, SAA solves the problem

$$\text{Problem } P_{m^*}: \quad \begin{array}{ll} \text{minimize} & f_{m^*}(x) \\ x \in \mathbb{X}, \end{array} \tag{1.2}$$

where $f_{m^*}(x) := F(m^*, x)$ is computed over a "fixed" sample of size m^*.

SAA is attractive in that Problem P_{m^*} becomes a *deterministic* optimization problem and SAA can bring to bear all of the advances in deterministic nonlinear programming methods [15] of the last few decades. The disadvantage, however, is the choice of the sample size m^*; results that identify m^* to guarantee some form of proximity of the estimator X_{m^*} to a solution to Problem P tend to be so conservative as to render implementation infeasible. There have been recent advances [82, 33, 13, 11, 12] that alleviate this issue to some degree but the question of sample size choice within SAA remains. See [56] for further discussion.

Stochastic Approximation (SA) [90, 20, 112, 84, 98, 57], the other popular method to solve SO problems of the kind (1.1), is more than six decades old. Virtually all SA type methods are of the form

$$X_{k+1} = X_k - \gamma_k \tilde{\nabla} f_{m^*}(X_k), \quad k = 1, 2, \ldots, \tag{SA}$$

where $\{\gamma_k\}$ is a sequence of positive constants, and $\hat{\nabla} f_{m^*}(X_k)$ is an estimate of the gradient of f at X_k calculated over a "random" sample of a *fixed* size m^*. (Within machine learning literature, SA is called stochastic gradient descent [69] when m^* is "small", and Batch Stochastic Gradient descent (BSG) method [105] when m^* is "large", relative to the size of the training sample set.)

SA, despite the existing body of literature, continues to be the subject of active study. This is because SA's success has been somewhat mixed due to the need to choose the gain sequence $\{\gamma_k\}$. The asymptotic theory lays down the conditions [19, 98, 57, 35, 88, 113, 92, 91, 96, 98, 100] on the sequence $\{\gamma_k\}$ to ensure that the resulting iterates $\{X_k\}$ are consistent and exhibit the fastest possible convergence rates under naïve Monte Carlo. Notably, theory for SA stipulates that the gain sequence $\{\gamma_k\}$ be chosen so that $\sum \gamma_k = \infty, \sum \gamma_k^2 < \infty$, that is, $\{\gamma_k\}$ should be chosen to converge to zero neither too fast nor too slow. (It is also known [70, 98] that the choice $\gamma_k = \frac{2}{k} H^{-1}$ retrieves the best possible asymptotic covariance matrix, where H is the differential of ∇f at the solution x^*.) While theory for SA's optimal convergence is attractive, it has proven of limited practical value. The prescribed optimal family of gain sequences is too large. Whether a chosen gain sequence (guaranteeing the fastest convergence rates) will perform well for a specific problem, is subject to chance; and, while it is possible to tune the gain sequence and "make" SA perform well for a given problem, or even a class of problems, formulating rules that automatically tune the gain sequence to achieve good finite-time performance has remained elusive. This opinion is supported by continuing efforts (after six decades of existing literature) to devise rules that either dynamically choose the gain sequence [54, 23, 22, 118] based on the

observed history of algorithm evolution, or by mitigating the effect of the gain sequence [38, 75, 85].

1.2 Sampling Controlled Stochastic Recursion

The paradigm we propose in this research can be viewed as being in-between SA and SAA. Like SAA, we would like to exploit advances in deterministic non-linear programming; like SA, we would like our solution to be in a simple recursive form but without the need to choose parameter sequences. So, we ask: why not use a recursion that would be used in a situation where all functions appearing within Problem P can be observed without error, but then replace the objects appearing within such a recursion by their Monte Carlo counterparts? An example serves to illustrate such a technique best. Consider the basic quasi-Newton recursion

$$x_{k+1} = x_k - \bar{\bar{H}}^{-1}(x_k)\tilde{\nabla} f(x_k), \quad k = 1, 2, \ldots, \tag{1.3}$$

used to find a local minimum of f, where $\bar{\bar{H}}(x)$ and $\tilde{\nabla} f(x)$ are the Hessian and gradient (deterministic) approximations of the true Hessian and gradient of the function $f : \mathbb{R}^d \to \mathbb{R}$ at the point x [99]. It is worth noting that $\bar{\bar{H}}(\cdot)$ and $\tilde{\nabla} f(\cdot)$ are "deterministic", and could be, for example, approximations obtained through appropriate finite-differencing of the function f at a set of points. If only "noisy" simulation-based estimates of f are available, then a reasonable adaptation of (1.3) is to use the recursion

$$X_{k+1} = X_k - \bar{\bar{H}}^{-1}(m_k, X_k)\tilde{\nabla} f(m_k, X_k) \quad k = 1, 2, \ldots, \tag{1.4}$$

where $\tilde{\nabla} f(m_k, x)$ and $\bar{\bar{H}}(m_k, x)$ are simulation-based estimates of $\tilde{\nabla} f(x)$ and $\bar{\bar{H}}(x)$ respectively at x, constructed using *estimated* function values, and $\{m_k\}$ is the sequence of the iterative sample size.

Our main interest in this dissertation is a generalized version of (1.4), called Sampling Controlled Stochastic Recursion SCSR (see Chapter 3). To reiterate, SCSR rests on the general philosophy that to solve Problem P, one should use a recursion that would be appropriate in the deterministic context, while replacing any existing objects within the recursion with their Monte Carlo counterparts. This broad idea sounds general and attractive in principle. However, the SCSR framework raises numerous important issues needing detailed study, and such issues will form the essence of this dissertation.

For example, three questions pertinent to SCSR that we will address through the early chapters of this dissertation include the following:

$Q.1$ What is the formal structure of SCSR's iterations to solve Problem P, and what conditions ensure that the resulting iterates converge (in probability and almost surely) to a solution of Problem P?

$Q.2$ What is the convergence rate of the iterates resulting from SCSR, expressed as a function of the sample sizes and the speed of the underlying recursion used within SCSR?

$Q.3$ With reference to $Q.2$, are there specific SCSR recursions that guarantee a canonical rate, that is, the fastest achievable Monte Carlo convergence speed under generic sampling?

The questions $Q.1$–$Q.3$ seek to understand the relationship between the errors due to recursion and sampling that naturally arise in SCSR, and their implication to SO and SRFP algorithms. As we will demonstrate through our answers in Chapter 3, these errors are inextricably linked and fully characterizable. Furthermore, we will show that such characterization naturally leads to sampling regimes which, when combined with a deterministic recursion of a specified speed, result in specific SCSR convergence rates.

The answers to $Q.1$–$Q.3$ are still theoretical in the sense that they provide no practical guarantees.

$Q.4$ Given the answer to $Q.3$, can practical guidance on how much simulation effort should be expended *as the resulting recursion evolves through the search space* be provided? Specifically, is there a way to adativetively sample, that is, construct sampling strategies that are an exlicit function of algorithmic trajectories while ensuring optimal convergence rates characterized through the answer to $Q.3$?

$Q.5$ Can we illustrate the effectiveness of our answer to $Q.4$ through a non-trivial "real-world" application?

In answering $Q.4$, we propose a simple adaptive sampling realization of SCSR called Adaptive-SCSR that aims to dynamically balance sampling error that is inherent in the observed data with the estimated accuracy of its iterates. Adaptive-SCSR is designed to formalize the following loosely stated sampling philosophy: early in the search process, when the iterates are far away from the optimum, sample the objects within the recursion little since the current iterate is likely to be inferior and even a coarse estimation of the objective function will provide gains in the sense of advancement toward the solution; later in the search process, when the incumbent iterates are probably close to the solution, sample more since

any perceived gain in advancement toward the solution could be due to mischance stemming from the fact that the gradient of the objective function is close to zero. In Chapter 5, and in response to $Q.5$, we use Adaptive SCSR augmented with a line search procedure to solve a non-trivial parameter estimation problem. The algorithm is called "LIAS" and is implemented within the context of online classification of human brain signals towards constructing a Brain-Computer Interface (BCI). Extensive numerical results that we present in Chapter 5 and Appendix 6.2 provide evidence for the effectiveness of our ideas.

1.3 Document Organization

The rest of this document is organized as follows. In the ensuing section, we introduce much of the notation and convention used throughout the dissertation. This is followed by Chapter 2 where we clarify the effect of certain improvements to SA. Chapter 3 and 4 present the main results associated with SCSR and Adaptive-SCSR. Chapter 5 and Appendix 6.2 present results from applying LIAS on the BCI application, and eventually Appendix 6.2 is a listing of MATLAB source code for LIAS.

1.4 Notations and Conventions

We will adopt the following notation through out the dissertation.

(a) Throughout the document, "d" is an integer-valued number and denotes dimensionality, unless otherwise stated.

(b) If $x \in \mathbb{R}^d$ is a vector, then its components are denoted through $x := (x^{(1)}, x^{(2)}, \ldots, x^{(d)})$.

(c) We use $e_i \in \mathbb{R}^d$ to denote a unit vector whose ith component is 1 and whose every other component is 0, that is, $e_i(i) = 1$ and $e_i(j) = 0$ for $j \neq i$.

(d) For a sequence of random variables $\{X_k\}$, we say $X_k \xrightarrow{\text{p}} X$ if $\{X_k\}$ converges to X in probability; similarly, we say $X_k \xrightarrow{\text{d}} X$ to mean that $\{X_k\}$ converges to X in distribution, and finally $X_k \xrightarrow{\text{wp1}} X$ to mean that $\{X_k\}$ converges to X with probability one.

(e) $\text{dist}(x, B) = \inf\{\|x - y\| : y \in B\}$ denotes the Euclidean distance between a point $x \in \mathbb{R}^d$ and a set $B \subset \mathbb{R}^d$. Likewise $\text{diam}(B) = \sup\{\|x-y\| : x, y \in B\}$ denotes the diameter of the set $B \subset \mathbb{R}^d$.

(f) For a sequence of real numbers $\{a_k\}$, we say $a_k = o(1)$ if $\lim_{k\to\infty} a_k = 0$; and $a_k = O(1)$ if $\{a_k\}$ is bounded, i.e., $\exists c \in (0, \infty)$ with $|a_k| < c$ for large enough k. We say that $a_k = \Psi(1)$ if $a_k = O(1)$ but a_k is not $o(1)$.

(g) For a sequence of real numbers $\{a_k\}$, we say $a_k = o_p(1)$ if $a_k \xrightarrow{\text{p}} 0$ as $k \to \infty$; and $a_k = O_p(1)$ if $\{a_k\}$ is stochastically bounded, that is, for given $\epsilon > 0$ there exists $c(\epsilon) \in (0, \infty)$ with $\Pr\{|a_k| < c(\epsilon)\} > 1 - \epsilon$ for large enough k. We say that $a_k = \Psi_p(1)$ if $a_k = O_p(1)$ but a_k is not $o_p(1)$. For two sequences of real numbers $\{a_k\}, \{b_k\}$ we say $a_k \sim b_k$ if $\lim a_k/b_k = 1$.

The following definitions will help our exposition.

Definition 1.4.1. (ϵ-Efficieny.) *Denote $\Gamma_k := \sum_{i=1}^k m_i$ as the total samples used up till the kth iteration. A stochastic recursion defined by (1.4) ensures ϵ-efficiency in solving the problem (1.1), if, as $k \to \infty$,*

$$\Gamma_k^{1-\epsilon}\mathbb{E}[f(X_k) - f(x^*)] \;=\; O(1), \qquad \textit{for } 0 < \epsilon < 1. \tag{1.5}$$

Definition 1.4.2. *(Growth rate of a sequence.) A sequence $\{m_k\}_{k\geq 1}$ is said to exhibit Polynomial(λ_p, p) growth if $m_k = \lambda_p\, k^p, k = 1, 2, \ldots$ for some $\lambda_p, p \in (0, \infty)$; it is said to exhibit Geometric(c) growth if $m_{k+1} = c\, m_k, k = 1, 2, \ldots$ for some $c \in (1, \infty)$; and Exponential(λ_t, t) growth if $m_{k+1} = \lambda_t m_k^t, k = 1, 2, \ldots$ for some $\lambda_t, t \in (0, \infty)$.*

Definition 1.4.3. (Sub-Geometric rate function.) *A sequence $\{m_k\}_{k\geq 1}$, also as a function of k, is said to be a sub-geometric rate function [36] if*

$$\lim_{k\to\infty} \frac{log\ m_k}{k} = 0$$

We denote the family of these functions by s.Ge, and we say $\{m_k\}_{k\geq 1} \in s.Ge$.

Definition 1.4.4. *(A sequence increasing faster than another.) Let $\{m_k\}_{k\geq 1}$ and $\{\tilde{m}_k\}_{k\geq 1}$ be two positive-valued increasing sequences that tend to infinity. Then $\{m_k\}$ is said to increase faster than $\{\tilde{m}_k\}$ if $m_{k+1}/m_k \geq \tilde{m}_{k+1}/\tilde{m}_k$ for large enough k.*

Definition 1.4.5. *(Convergence rate of a sequence, and the uniform convergence rate of a family of sequences.) Consider a family of sequences $\mathcal{F}$ in $\mathcal{D}$ indexed by*

θ, with each element $\{z_k^\theta\}$ of $\mathcal{F}$ satisfying $\|z_k^\theta - z^*\| \to 0$ and $z_k \neq z^*$ for all but finitely many $z_k^\theta s$.

In what follows, (i) and (ii) are equivalent definitions of the convergence rate of an individual sequence $\{z_k^\theta\}$ to z^*, and (iii) defines the notion of equi-convergence rate of the family of sequences $\mathcal{F}$.

(i) $\{z_k^\theta\} \subset \mathcal{D}$ exhibits Linear(ℓ) convergence to $z^* \in \mathcal{D}$ if $\limsup_{k\to\infty} \frac{\|z_{k+1}^\theta - z^*\|}{\|z_k^\theta - z^*\|} = \ell \in (0,1)$; $\{z_k^\theta\}$ exhibits SuperLinear(q), $q \in (1,\infty)$ convergence to z^* if $\limsup_{k\to\infty} \frac{\|z_{k+1}^\theta - z^*\|}{\|z_k^\theta - z^*\|^q} = \lambda_q \in (0,\infty)$; $\{z_k^\theta\}$ exhibits SubLinear(s) convergence to z^* if $\limsup_{k\to\infty} k(1 - \frac{\|z_{k+1}^\theta - z^*\|}{\|z_k^\theta - z^*\|}) = s \in (0,\infty)$.

(ii) For $\ell \in (0,1)$, $\{z_k^\theta\} \subset \mathcal{D}$ exhibits Linear(ℓ) convergence to $z^* \in \mathcal{D}$ if for given small-enough $\epsilon > 0$, there exist $k_0^\theta(\epsilon)$ and $\Delta^\theta(\epsilon)$ such that if $k \geq k_0^\theta(\epsilon)$ and $\|z_k^\theta - z^*\| \leq \Delta^\theta(\epsilon)$, then $\|z_{k+1}^\theta - z^*\| \leq (\ell+\epsilon)\|z_k^\theta - z^*\|$ and an infinite number of z_k^θ satisfy $\|z_{k+1}^\theta - z^*\| \geq (\ell+\epsilon)\|z_k^\theta - z^*\|$; $\{z_k^\theta\}$ exhibits SuperLinear(q), $q \in (1,\infty)$ convergence to z^* if for given small-enough $\epsilon > 0$, there exist $k_0^\theta(\epsilon)$ and $\Delta^\theta(\epsilon)$ such that if $k \geq k_0^\theta(\epsilon)$ and $\|z_k^\theta - z^*\| \leq \Delta^\theta(\epsilon)$, then $\|z_{k+1}^\theta - z^*\| \leq (\ell+\epsilon)\|z_k^\theta - z^*\|^q$ and an infinite number of z_k^θ satisfy $\|z_{k+1}^\theta - z^*\| \geq (\ell+\epsilon)\|z_k^\theta - z^*\|^q$; $\{z_k^\theta\}$ exhibits SubLinear(s) convergence to z^* if for given small-enough $\epsilon > 0$, there exist $k_0^\theta(\epsilon)$ and $\Delta^\theta(\epsilon)$ such that if $k \geq k_0^\theta(\epsilon)$ and $\|z_k^\theta - z^*\| \leq \Delta^\theta(\epsilon)$, then $\|z_{k+1}^\theta - z^*\| \leq (1 - \frac{s-\epsilon}{k})\|z_k^\theta - z^*\|$ and an infinite number of z_k^θ satisfy $\|z_{k+1}^\theta - z^*\| \leq (1 - \frac{s+\epsilon}{k})\|z_k^\theta - z^*\|$.

(iii) The family of sequences $\mathcal{F}$ is said to exhibit equi-Linear(ℓ), equi-SuperLinear(q), and equi-SubLinear(s) convergence if each sequence $\{z_k^\theta\} \in \mathcal{F}$ exhibits Linear(ℓ), SuperLinear(q), and SubLinear(s) convergence respectively, and in each case the constants $k_0^\theta(\epsilon)$ and $\Delta^\theta(\epsilon)$ appearing in (ii) can be chosen independent of the index θ.

Chapter 2

Stochastic Approximation (SA)

The broad setting of this chapter is stochastic approximation (SA), the famous iteration originally introduced by [90] as a method to identify the zero of a function. As mentioned in Chapter 1, SA uses a Newton-type recursion to converge, where the point estimators across replications are constructed by (re-)sampling a fixed number of samples m^*. The modern version of Robbins and Monro's SA iteration usually takes the form

$$X_k = X_{k-1} - \gamma_k \bar{\bar{H}}_k^{-1} \tilde{h}(X_{k-1}), k = 1, 2, \ldots, \tag{2.1}$$

where $\{\gamma_k\}$ is a positive sequence converging to 0. The iteration has been widely used in both the optimization and root-finding contexts. When used in the root-finding context, the objective of the iteration in (2.1) is identifying a zero of the function $h(\cdot)$, while the estimator $\tilde{h}(\cdot)$ provides noisy observations of the function $h(\cdot)$. When used in the optimization context, the objective of the iteration in (2.1) is identifying a stationary point of a real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ that is "observed" using an unbiased estimator $F(\cdot)$. In such a case, the quantity $\tilde{h}(X_{k-1})$ appearing in (2.1) estimates the gradient of the function $f(\cdot)$ at the point X_{k-1}, and is usually constructed using forward or central differencing [98]. In the optimization setting, $\bar{\bar{H}}_k$ then estimates the Hessian (matrix of second derivatives) of the function $f(\cdot)$ at the point X_{k-1}, and is again calculated using some form of differencing [98]. The SA iteration as stated in (2.1) is for the unconstrained context. Extending it to tackle problems with a (deterministically) constrained feasible region is usually done by performing an appropriate projection operation back into the feasible region whenever the iterates drift outside the feasible region.

SA is arguably the most popular current method of solving continuous local optimization and root-finding problems when the functions involved can only be estimated (and the constraints are known and deterministic). Owing to its simplicity,

its interpretation as the natural stochastic analogue of Newton's method, and its attractive asymptotic properties, SA has seen a tremendous amount of application [57]. A lot has been written on the topic, and the finite-time and infinite-time behavior of the recursion in (2.1) is well-understood. (There are several books that will serve as good entry points into this literature, e.g., [57, 21, 98, 112].)

Despite SA's enduring popularity and the six decades of research supporting its advance, the prevailing opinion is that choosing the "gain sequence" $\{\gamma_k\}$ to ensure robust and efficient SA performance is challenging [96, 100, 23, 22, 84, 85]. In other words, while it is possible to tune the gain sequence and "make" SA perform well for a given problem, or even a class of problems, formulating rules that *automatically* tune the gain sequence to achieve good finite-time performance is still an open problem (albeit loosely defined). This opinion is also supported by continuing efforts to devise rules that either dynamically choose the gain sequence [23, 22, 117] based on the observed history of algorithm evolution, or by mitigating the effect of the gain sequence [75, 85].

Of particular interest in this chapter are two advances that have been crucial milestones in SA's history. The first is what is popularly called "Polyak Averaging" [88] which involves the simple idea of averaging SA's iterates. To elaborate, various authors [30, 34, 40] prior to 1997 had shown that the best possible convergence rate of SA's iterates (to the correct solution) is $O(1/\sqrt{k})$, achieved when the gain sequence $\gamma_k = O(1/k)$. (Rigorously, this implies that when $\gamma_k = N/k$ and N is larger than half the inverse of the smallest eigen value of the function f's Hessian at the solution, the iterates can be shown to satisfy $\sqrt{k}(X_k - x^*) \xrightarrow{\mathrm{d}} N(0, V)$ where x^* is a solution to the problem, V is a covariance matrix.) While this result is useful, finite-time performance considerations suggested using step sizes that were larger, i.e., converged slower, than the $O(1/k)$ suggested by asymptotic performance considerations. The dilemma was that choosing a slowly converging gain sequence, e.g., $\gamma_k = O(1/k^\alpha), \alpha \in (0, 1)$, while often producing better finite-time performance, degraded SA's asymptotic convergence rate. [88], and simultaneously [43], provided an elegant solution for this dilemma. [88] showed that SA can be executed on two timescales to enjoy good finite-time performance while not sacrificing asymptotic performance. Specifically, he suggested executing SA on the "fast timescale" $X_k = X_{k-1} - \gamma_k \tilde{h}(X_{k-1}), \gamma_k = O(k^{-\alpha}), \alpha \in (0, 1)$ and then averaging the iterates $X_k, k = 1, 2, \ldots$ offline to get $Y_k = k^{-1} \sum_{i=1}^{k} X_i$. He demonstrated the remarkable result that, under certain conditions, such a two timescale averaging produced the averaged iterates $\{Y_k\}$ having the best possible convergence rate $O(1/\sqrt{k})$. (He also showed that the iterates X_k attain the degraded convergence rate of $O(\gamma_k)$.) Polyak's paper was written within the context of root-finding. This was extended to the optimization context by [35].

The second milestone of interest in this chapter is the efficient use of derivatives within SA. It is clear from the corresponding literature in the deterministic context that knowledge of the Jacobian matrix of derivatives $H : \mathbb{R}^d \to \mathbb{R}^d$ of the function $h(\cdot)$ can help immensely with efficient searching within the solution space. This prompted modifying the original SA iteration to incorporate derivative estimates to obtain the modified iteration $X_k = X_{k-1} - \gamma_k \bar{\bar{H}}_k^{-1} \tilde{h}(X_{k-1})$. While this sounds reasonable, the main issue with this approach turns out to be the computation involved in estimating the derivative estimate $\bar{\bar{H}}_k$. For instance, if one indulged in estimating every entry of $\bar{\bar{H}}_k$ using a method such as forward differences, this would involve $O(d^2)$ simulations just to obtain the estimated derivative at the incumbent point. This led [98] to investigate more efficient methods to obtain a derivative estimate. While a lot has been written on this particular issue, the crux of Spall's work is that with just $O(d)$ simulations, it is possible to obtain derivative estimates that do not degrade the asymptotic convergence rate of the SA iteration. In other words, using just a crude calculation of the derivative $\bar{\bar{H}}_k$, one could potentially enjoy the benefits of better finite-time performance without sacrificing the asymptotic convergence rate of $O(1/\sqrt{k})$. The method is called Simultaneous Perturbation SA (SPSA), and is showed to be asymptotically efficient when $\gamma_k = O(1/k)$.

As discussed in Polyak averaging idea however, using small step size of the order $1/k$ often results in stalling behavior of SA. The same situation holds even when using derivative estimates within SA recursion. Accordingly [98] suggested using larger than usual step sizes in practice to get a better performance (i.e. $\gamma_k = O(k^{-\alpha}), \alpha \in (0, 1)$), while efficiency results hold with $\gamma_k = O(1/k)$. On the other hand, Polyak averaging is shown to be performing well with larger step sizes only in situation where the iterates oscillate around the true solution. Otherwise the averaged iterate may even hurt the accuracy of the incumbent solution. But the question is, how can we preserve the accuracy of the averaged iterate in general and improve the direction of search while taking large steps towards the true solution?

The idea in this chapter, is to tackle this issue and to fill the gaps in the two approaches (i.e. not efficient iterates when using SPSA with large step size, and not well-behaved averaged iterates due to improper search direction) simultaneously, by proposing a "joint scenario". Accordingly we investigate the effect of averaging on SPSA, and ask if SPSA retains the fastest possible $O(1/\sqrt{k})$ convergence rate even when using larger than usual step sizes. (We have found no evidence of any analysis in the literature that incorporates both of these ideas. Even the most recent literature on this topic [71, 75, 117] do not incorporate estimated Hessians into the SA iteration, most likely due to computational considerations.)

Towards exploring this idea, we ask the following two questions.

Q.1 When Polyak averaging and derivative estimates are included within the SA iteration, what conditions ensure that the averaged iterates retain the $O(1/\sqrt{k})$ convergence?

Q.2 Can anything be said about the convergence characteristics of SA's faster timescale iterates?

We start by answering $Q.2$. We demonstrate that, amongst other conditions, if the sequence $\{\gamma_k \bar{\bar{H}}_k^{-1}\}$ satisfies a certain stochastic-matrix analogue of regularly varying sequences, and the Hessian estimator $\bar{\bar{H}}_k$ is consistent in a certain precise sense, the faster timescale iterates $\{X_k\}$ converge in mean square at the rate $O(\gamma_k \log k)$. This rate is slightly slower than that obtained without the Hessian estimator, The condition we impose on $\{\gamma_k \bar{\bar{H}}_k^{-1}\}$ is not entirely new and is closely related to conditions established in [88] and [71].

In answering $Q.1$, we show that conditions similar to that used in answering $Q.2$ ensure that the slower timescale sequence $\{Y_k\}$ retains the $O(1/\sqrt{k})$ convergence rate. This should come as no surprise to the reader and should be seen simply as theoretical confirmation of what seems intuitively clear.

The remainder of this chapter is organized as follows. In Section 2.1.1, we outline the sufficient conditions to retrieve the rate at which the fast timescale sequence converges to the root which is established in Section 2.1.2. In Section 2.1.3 we establish the almost-sure convergence of the averaged iterates within the SA iteration together with a simple extension of the SA iteration for relaxing the need to pre-specify the gain sequence. Concluding remarks are made finally in Section 2.2.

2.1 Main Results

In everything that follows, the SA iteration of interest is the two timescale recursion given by

$$\begin{aligned} X_k &= X_{k-1} - \Lambda_k \tilde{h}_k; \\ Y_k &= (1 - \frac{1}{k+1}) Y_{k-1} + \frac{1}{k+1} X_k; \end{aligned} \tag{2.2}$$

where $\Lambda_k = \gamma_k \bar{\bar{H}}_k^{-1}$, $\bar{\bar{H}}_k$ is a consistent estimator of the derivative of $h(\cdot)$ at X_{k-1}, and $\{\gamma_k\}$ is a positive sequence converging to 0. Also, as is common SA settings, we assume that $\tilde{h}_k = h(X_{k-1}) + \varepsilon_k$ is the noisy observation of the function h at the point X_{k-1}, where ε_k is a random disturbance. We emphasize that for purposes of

this chapter, our interest will be limited to the context of root-finding within the unconstrained context.

2.1.1 Assumptions

C.1 Suppose that the solution to the vector equation $h(x) = 0$ is x^*. We assume the existence of $\eta > 1$ and a neighborhood $\mathcal{N}(x^*)$ of x^* such that $h(x) = H(x - x^*) + O(\|x - x^*\|^\eta)$ for $x \in \mathcal{N}(x^*)$, where the matrix $-H$ is Hurwitz.

C.2 There exists $\rho_1 > 0$ for which $\mathbb{E}\|\tilde{h}(x)\|^2 \leq \rho_1(1 + \|x - x^*\|^2)$.

C.3 For the consistent estimator $\bar{\bar{H}}_k$, we assume the boundedness of moments, i.e., the existence of a positive ρ such that $E(\|\bar{\bar{H}}_k^{-1}\|^2) \leq \rho$.

C.4 For each $k \geq 1$ and all x there exists $\rho_2 > 0$ not dependent on k and x such that
$(x - x^*)^T \bar{h}_k(x) \geq \rho_2 \|x - x^*\|^2$, where $\bar{h}_k(x) = \bar{\bar{H}}_k^{-1} \tilde{h}(x_k)$.

C.5 (a) $\lim_{k\to\infty} k(I - \Lambda_k^{-1}\Lambda_{k+1}) \xrightarrow{\text{p}} \alpha I$, $1/2 < \alpha < 1$, $\Lambda_k = \gamma_k \bar{\bar{H}}_k^{-1}$, and $\gamma_k \to 0$, $\gamma_k > 0$.

(b) $\lim_{k\to\infty} \log k / k\gamma_k \to 0$ and $\displaystyle\sum_{k=1}^{\infty} \frac{(\gamma_k \log k)^{\frac{\eta}{2}}}{\sqrt{k}} < \infty$ for $\eta > 1$.

C.6 $E(\varepsilon_{k+1}|\mathcal{F}_k) = 0$ where $\mathcal{F}_k = \{\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{k-1}, X_k, \bar{\bar{H}}_k\}$ and there exists a non-random, positive definite matrix Γ such that $\lim_{k\to\infty} E(\varepsilon_{k+1}\varepsilon_{k+1}^T|\mathcal{F}_k) = \Gamma$ almost surely.

Assumptions similar to *C*.1 and *C*.2 are common within the literature on Polyak averaging. For instance, [88], [23] and [71] impose similar conditions. The assumptions *C*.3 and *C*.4 are prevalent in the SA literature that uses an estimated derivative within the recursion. For example, [97] makes this assumption. The main condition of interest is *C*.5(a) and it should be seen as the matrix-analogue of the original condition introduced by [88]. Assuming consistency of $\bar{\bar{H}}_k$, this assumption implies that γ_k is a regularly varying sequence as originally introduced by [44]. Interestingly, *C*.5 implies that $\sum_{i=1}^k \gamma_i \to \infty$ and $\sum_{i=1}^k \gamma_i^2 < \infty$ as $k \to \infty$, conditions that are routinely assumed (directly) within the SA literature.

2.1.2 Behavior of the Fast Timescale Iterates

In this section, we present a result that characterizes the rate at which the fast timescale sequence $\{X_k\}$ converges to the root x^*.

Theorem 2.1.1. *Let assumptions C.1 – C.5 hold, and let $\rho_2^2 < \rho\rho_1$. Then the mean squared error $mse(X_k, x^*)$ of X_k with respect to x^* satisfies $mse(X_k, x^*) = O(\gamma_k \log k)$.*

Proof. Let $A_{k+1} = \|X_{k+1} - x^*\|^2$. Then

$$\begin{aligned} A_{k+1} &= \|X_k - x^* - \Lambda_k \tilde{h}(X_k)\|^2 \\ &= \|X_k - x^*\|^2 - 2\gamma_k (X_k - x^*)^T \bar{h}(X_k) + \gamma_k^2 \|\bar{h}(X_k)\|^2. \end{aligned} \tag{2.3}$$

By assumptions $C.2$ and $C.3$, we get

$$E[\|\bar{h}(X_k)\|^2 | X_k] \leq \rho\rho_1(1 + A_k);$$

and in view of assumption $C.4$ we have

$$E[-(X_k - x^*)^T \bar{h}(X_k) | X_k] \leq -\rho_2 A_k.$$

Taking expectations on both sides in (2.3) after conditioning on X_k we get

$$E[A_{k+1} | X_k] \leq A_k(1 - 2\rho_2\gamma_k + \rho\rho_1\gamma_k^2) + \rho\rho_1\gamma_k^2. \tag{2.4}$$

If we now let $b_k := E[A_{k+1}]$, we get

$$b_k \leq b_1 \prod_{i=1}^{k} p_i + \sum_{i=2}^{k-1} \prod_{j=i+1}^{k} q_i p_j + q_k := u_k,$$

where $p_i = (1 - 2\rho_2\gamma_i + \rho\rho_1\gamma_i^2), q_i = \rho\rho_1\gamma_i^2$. Since we have chosen ρ, ρ_1, ρ_2 in such a way that $\rho\rho_1 > \rho_2^2 > 0$ for all i, p_i and q_i are positive.

Define $k_0 := \sup\{k \geq 1 \ : \ \rho_2 < 2\rho\rho_1\gamma_k, \ \rho_2 < 2\rho\rho_1\alpha\frac{\gamma_k}{k}, k\gamma_k < \frac{2\alpha}{\rho_2}, \text{ and } \log k - 1 < \frac{2\rho\rho_1}{\rho_2}\} + 1$ and choose c large enough to satisfy the following

$$\frac{u_{k_0+1}}{\gamma_{k_0} \log k_0} \leq c.$$

Then one can see by induction that for all $k \geq 1$, $b_{k+1} \leq c\gamma_k \log k$, where

$$c = \max\{1, \max_{1 \leq k \leq k_0} \{\frac{u_{k+1}}{\gamma_k \log k}\}\}.$$

□

Theorem 2.1.1 asserts that the fast timescale iterates converge at the rate $O(\gamma_k \log k)$. Since the sequence $\{\gamma_k\}$ converges slower than $O(1/k)$, this points to a degraded rate of convergence for the fast timescale iterates.

2.1.3 Weak Convergence of the Slow Timescale Iterates

Theorem 2.1.2 demonstrates that the averaged iterates within the SA iteration in (2.2) attain the best possible convergence rate in a weak sense.

Theorem 2.1.2. *Under assumptions C.1–C.6, we have*

(*i*) $\sqrt{k}(Y_k - x^*) \xrightarrow{d} N(0, H^{-1}\Gamma[H^{-1}]^T)$, *where* H *represents the Jacobian of* $h(z)$ *at* $z = x^*$;

(*ii*) $Y_k - x^* \to 0$ *almost surely.*

Proof of (i). Let $\Delta_k = X_k - x^*$ and $\bar{\Delta}_k = Y_k - x^*$. Then

$$\begin{aligned} X_k &= X_{k-1} - \Lambda_k(H\Delta_{k-1} + \varepsilon_k); \\ \Delta_{k-1} &= H^{-1}\Lambda_k^{-1}(X_{k-1} - X_k) - H^{-1}\varepsilon_k; \\ \Delta_k &= H^{-1}\Lambda_{k+1}^{-1}(X_k - X_{k+1}) - H^{-1}\varepsilon_{k+1}. \end{aligned}$$

On the other hand

$$\begin{aligned} Y_k &= (1 - \frac{1}{k+1})Y_{k-1} + \frac{1}{k+1}X_k; \\ Y_k - x^* &= (1 - \frac{1}{k+1})(Y_{k-1} - x^*) + \frac{1}{k+1}(X_k - x^*); \\ \bar{\Delta}_k &= (1 - \frac{1}{k+1})\bar{\Delta}_{k-1} + \frac{1}{k+1}\Delta_k. \end{aligned}$$

Set $\prod_{j=k+1}^{k}(1 - \frac{1}{j+1}) = 1$. So we get

$$\begin{aligned} \bar{\Delta}_k &= \prod_{j=1}^{k}(1 - \frac{1}{j+1})\Delta_0 \\ &\quad + \sum_{i=1}^{k}\prod_{j=i+1}^{k}(1 - \frac{1}{j+1})\frac{1}{i+1}H^{-1}\Lambda_{i+1}^{-1}(X_i - X_{i+1}) \\ &\quad - \sum_{i=1}^{k}\prod_{j=i+1}^{k}(1 - \frac{1}{j+1})\frac{1}{i+1}H^{-1}\varepsilon_{i+1}. \end{aligned}$$

Let

$$R_{k+1}^1 = \prod_{j=1}^{k}(1-\frac{1}{j+1})\Delta_0;$$
$$R_{k+1}^2 = \sum_{i=1}^{k}\prod_{j=i+1}^{k}(1-\frac{1}{j+1})\frac{1}{i+1}H^{-1}\Lambda_{i+1}^{-1}(X_i - X_{i+1});$$
$$R_{k+1}^3 = \sum_{i=1}^{k}\prod_{j=i+1}^{k}(1-\frac{1}{j+1})\frac{1}{i+1}H^{-1}\varepsilon_{i+1}.$$

Note that as $k \to \infty$, $\sqrt{k}R_{k+1}^1 = \frac{\sqrt{k}}{k+1}\Delta_0 \overset{\text{wp1}}{\to} 0$.

Also by [88], $\sqrt{k}R_{k+1}^3 \overset{\text{d}}{\to} N(0, H^{-1}\Gamma[H^{-1}]^T)$. So we just need to prove that $\sqrt{k}R_{k+1}^2 \overset{P}{\to} 0$.

$$\begin{aligned}
R_{k+1}^2 &= \sum_{i=1}^{k}\prod_{j=i+1}^{k}(1-\frac{1}{j+1})\frac{1}{i+1}H^{-1}\Lambda_{i+1}^{-1}(X_i - X_{i+1}),\\
&= \frac{1}{k+1}\sum_{i=1}^{k}H^{-1}\Lambda_{i+1}^{-1}[(X_i - x^*) - (X_{i+1} - x^*)],\\
&= \frac{1}{k+1}H^{-1}\Lambda_2^{-1}(X_1 - x^*) \ + \ \frac{1}{k+1}\sum_{i=2}^{k}H^{-1}\Lambda_{i+1}^{-1}(X_i - x^*)\\
&\quad -\frac{1}{k+1}\sum_{i=1}^{k-1}H^{-1}\Lambda_{i+1}^{-1}(X_{i+1} - x^*) + \frac{1}{k+1}H^{-1}\Lambda_{k+1}^{-1}(X_{k+1} - x^*).
\end{aligned}$$

Since we have

$$\frac{1}{k+1}\sum_{i=1}^{k-1}H^{-1}\Lambda_{i+1}^{-1}(X_{i+1} - x^*) = \frac{1}{k+1}\sum_{i=2}^{k}H^{-1}\Lambda_i^{-1}(X_i - x^*),$$

we can write

$$\begin{aligned}
R_{k+1}^2 &= \frac{1}{k+1}\sum_{i=2}^{k}[H^{-1}\Lambda_{i+1}^{-1}(I - \Lambda_{i+1}\Lambda_i^{-1})(X_i - x^*)]\\
&\quad -\frac{1}{k+1}H^{-1}\Lambda_{k+1}^{-1}(X_{k+1} - x^*) \ + \ \frac{1}{k+1}H^{-1}\Lambda_2^{-1}(X_1 - x^*).
\end{aligned}$$

In view of Theorem 2.1.1 and assumption $C.4$, we then get:

$$\begin{aligned} R_{k+1}^2 &= \frac{1}{k+1}\sum_{i=2}^{k}[H^{-1}\Lambda_{i+1}^{-1}o_p(\frac{1}{i+1})o(\sqrt{\gamma_i \log i})] \\ &\quad -\frac{1}{k+1}H^{-1}\Lambda_{k+1}^{-1}o(\sqrt{\gamma_{k+1}\log k+1}) \ + \ o(\frac{1}{\sqrt{k+1}}), \\ &= \frac{1}{k+1}\sum_{i=2}^{k} o_p[\frac{1}{i+1}\sqrt{\frac{\gamma_i \log i}{\gamma_{i+1}^2}}] - o_p[\frac{1}{k+1}\sqrt{\frac{\gamma_{k+1}\log k+1}{\gamma_{k+1}^2}}] \ + \ o(\frac{1}{\sqrt{k+1}}). \end{aligned}$$

Hence

$$\begin{aligned} \sqrt{k}R_{k+1}^2 &= \frac{1}{\sqrt{k+1}}\sum_{i=2}^{k} o_p(\frac{1}{\sqrt{i+1}}) - o_p(1) \ + \ o(1), \\ &= o_p(1). \end{aligned}$$

Proof of part (i). (Assume that the underlying function h is nonlinear)

By assumption $C.1$ we get:

$$\begin{aligned} \bar{\Delta}_k &= \prod_{j=1}^{k}(1-\frac{1}{j+1})\Delta_0 \\ &\quad +\sum_{i=1}^{k}\prod_{j=i+1}^{k}(1-\frac{1}{j+1})\frac{1}{i+1}H^{-1}\Lambda_{i+1}^{-1}(X_i - X_{i+1}) \\ &\quad +\sum_{i=1}^{k}\prod_{j=i+1}^{k}(1-\frac{1}{j+1})\frac{1}{i+1}H^{-1}o(\|X_i - x^*\|^{\eta}) \\ &\quad -\sum_{i=1}^{k}\prod_{j=i+1}^{k}(1-\frac{1}{j+1})\frac{1}{i+1}H^{-1}\varepsilon_{i+1}. \end{aligned}$$

Let

$$\tilde{R}_{k+1} = \sum_{i=1}^{k}\prod_{j=i+1}^{k}(1-\frac{1}{j+1})\frac{1}{i+1}H^{-1}o(\|X_i - x^*\|^{\eta}).$$

Thus, we are only yet to prove that $\sqrt{k}\tilde{R}_{k+1} \to 0$ as $k \to \infty$. By Theorem 2.1.1, we have

$$\begin{aligned} \sqrt{k}\tilde{R}_{k+1} &= \frac{1}{\sqrt{k}}\sum_{i=1}^{k} o((\gamma_i \log i)^{\frac{\eta}{2}}), \\ &= \frac{1}{\sqrt{k}}\sum_{i=1}^{k}\sqrt{i}\, o(\frac{(\gamma_i \log i)^{\frac{\eta}{2}}}{\sqrt{i}}). \end{aligned}$$

The claim then follows by assumption $C.5$(b) and Kronecker's lemma.

Proof of part (ii).

$$R^1_{k+1} = \prod_{j=1}^{k}(1 - \frac{1}{j+1})\Delta_0 = \frac{\Delta_0}{k+1}$$

and so $R^1_{k+1} \to 0$ as $k \to \infty$.

$$R^2_{k+1} = \frac{1}{k+1}\sum_{i=2}^{k} o_p[\frac{1}{i}\sqrt{\frac{\gamma_i \log i}{\gamma^2_{i+1}}}] - \frac{1}{k+1}\gamma^{-1}_{k+1}H^{-1}\bar{\bar{H}}_{k+1}o(\sqrt{\gamma_{k+1}\log k+1}) + o(\frac{1}{\sqrt{k}}),$$

and by Cesaro summability [16] and $C.5$(a), $R^2_{k+1} \to 0$ as $k \to \infty$. Finally,

$$R^3_{k+1} = \frac{1}{k+1}\sum_{i=1}^{k} H^{-1}\varepsilon_{i+1},$$

and so by the strong law of large numbers [16] we get $R^3_{k+1} \to 0$ as $k \to \infty$.

□

In conclusion and as a further step in the direction of completely relaxing the need to pre-specify the gain sequence, we now propose a simple extension of the SA iteration considered thus far.

$$X_{j+1} = X_j - \Lambda_{t_j}\tilde{h}(X_j), j = 1, 2, \ldots, \tag{2.5}$$

where $t_j := \text{Min}\{t : N_t \geq j\}$ for $j = 1, 2, \ldots$, $\Lambda_{t_j} = \gamma_{t_j}\bar{\bar{H}}^{-1}_j$, and $\bar{\bar{H}}_j$, $\tilde{h}(X_j)$ are as defined in (2.2). It can be seen that the iteration in (2.5) is constructed to facilitate designing heuristics that dynamically change the step sizes based on observed history of the SA iteration.

The following theorem establishes the asymptotic efficiency of (2.5) under suitable conditions.

Theorem 2.1.3. *Let $(N_t)_{t\geq 0}$ be an increasing sequence of random variables with $k_0 = 0$ and let $\Delta_t = N_t - N_{t-1}$.*

(*I*) *Suppose assumption $C.2 - C.4$ and $C.6$ hold true. Further suppose that Δ_t is uniformly bounded for all t. If the gain sequence $\{\gamma_t\}$ satisfies $\sum_{t=1}^{\infty}\gamma_t = \infty$ and $\sum_{t=1}^{\infty}\gamma^2_t < \infty$, the iteration in (2.5) converges to x^* a.s.*

(II) Moreover, consider the following two time scale SA algorithm:

$$X_{k+1} = X_k - \Lambda_{t_k}\tilde{h}(X_k);$$
$$\bar{X}_{k+1} = (1 - \frac{1}{n+2})\bar{X}_k + \frac{1}{n+2}X_{k+1}; \tag{2.6}$$

Suppose conditions $C.1 - C.6$ hold, and for a $\zeta > 0$, $\frac{t_k}{k} \overset{wp1}{\to} \zeta$ as $k \to \infty$. Then we have

(i) $\sqrt{k}(\bar{X}_k - x^*) \overset{d}{\to} N(0, H^{-1}\Gamma[H^{-1}]^T)$;

(ii) $\bar{X}_k - x^* \to 0$ *almost surely.*

2.2 Concluding Remarks

Polyak averaging attempts to ensure efficiency of SA while a "large" step size is employed in the recursion to improve finite time behavior. On the other hand, the presence of derivative estimates within the recursion helps to improve the search direction through SPSA method. While Polyak averaging does not always satisfy a good search direction, SPSA is lacking efficiency when using larger than usual step sizes. The two main results presented in this chapter, characterize the behavior of SA's iterates under a "joint scenario", and show that the averaged iterates, upgraded with derivative estimates, retain the best possible convergence rate under mild stipulations on the quality of the derivative estimates and the gain sequence. Our treatment in this chapter was limited to the context of root-finding, but extensions to the optimization context seem evident.

Chapter 3

Sampling Controlled Stochastic Recursions (SCSR)

We consider the question of sampling within algorithmic recursions that involve quantities needing to be estimated using a stochastic simulation. While much of what we say in this chapter applies more widely, the prototypical example setting is Simulation Optimization (SO) [47], where an optimization problem is to be solved using only a stochastic simulation capable of providing estimates of the objective function and constraints at a requested point. Another closely related example setting is the Stochastic Root Finding Problem (SRFP) [84, 81, 86], where the zero of a vector function is sought, with only simulation-based estimates of the function involved. SO problems and SRFPs have recently generated great attention owing to their flexible problem statements. Specifically, instead of stipulating that the functions involved in the problem statement be known exactly or in analytic form, SO problems and SRFPs allow implicit representation of functions through a stochastic simulation, thereby allowing the embedding of virtually any level of complexity. Such flexibility has resulted in adoption across widespread application contexts. A few examples are logistics [48, 50, 8], healthcare [1, 32, 29], epidemiology, and vehicular-traffic systems. An entire track has been devoted to simulation optimization and its applications in recent years of the Winter Simulation Conference (WSC).

A popular and reasonable solution paradigm for solving SO problems and SRFPs is to simply mimic what a solution algorithm might do within a deterministic context, after estimating any needed function and derivative values using the available stochastic simulation. An example serves to illustrate such a technique best. Con-

sider the basic quasi-Newton recursion

$$x_{k+1} = x_k - \bar{\bar{H}}^{-1}(x_k)\tilde{\nabla} f(x_k), \tag{3.1}$$

used to find a local minimum of a twice-differentiable real-valued function $f : \mathbb{R}^d \to \mathbb{R}$, where $\bar{\bar{H}}(x)$ and $\tilde{\nabla} f(x)$ are the Hessian and gradient (deterministic) approximations of the true Hessian and gradient of the function $f : \mathbb{R}^d \to \mathbb{R}$ at the point x. It is worth noting that $\bar{\bar{H}}(\cdot)$ and $\tilde{\nabla} f(x)$ are "deterministic", and could be, for example, approximations obtained through appropriate finite-differencing of the function f at a set of points. If only "noisy" simulation-based estimates of f are available, then a reasonable adaptation of (3.1) might be to use the recursion

$$x_{k+1} = x_k - \bar{\bar{H}}^{-1}(m_k, x_k)\tilde{\nabla} f(m_k, x_k) \tag{3.2}$$

where $\tilde{\nabla} f(m_k, x)$ and $\bar{\bar{H}}(m_k, x)$ are simulation-based estimates of $\tilde{\nabla} f(x)$ and $\bar{\bar{H}}(x)$, constructed using *estimated* function values. The simulation effort m_k is general and might represent the number of simulation replications in the case of terminating simulations or the simulation run length in the case of non-terminating simulations [58].

Important questions arise within the context of using recursions such as (3.2). Specifically, the iterates resulting from (3.2) incur two types of error: (i) recursion-error, incurred due to the structure of (3.2); and (ii) sampling-error, incurred due to the fact that the exact value of the function at any point x is unknown and needs to be estimated using stochastic sampling. Since sampling serves to reduce only the latter error, "too much" sampling will likely be inefficient. Likewise, "too little" sampling will also be inefficient since the sampling error will then tend to dominate the recursion error. (In fact, unlike too much sampling which affects only efficiency, we will show that too little sampling may threaten even consistency, that is, it may even cause iterates to not converge to the correct solution.) Intuition thus dictates that the interplay between the errors in (i) and (ii) should be characterized and "balanced" if the resulting iterates are to evolve efficiently towards the correct solution.

Accordingly, the questions we answer in this chapter pertain to the (simulation) sampling effort expended within recursions such as (3.2). Our interest is a generalized version of (3.2) called Sampling Controlled Stochastic Recursion SCSR, and within which we ask the following:

- $Q.1$ what sampling rates in SCSR ensure that the resulting iterates are strongly consistent?
- $Q.2$ what is the convergence rate of the iterates resulting from SCSR, expressed as a function of the sample sizes and the speed of the underlying deterministic recursion?

$Q.3$ with reference to $Q.2$, are there specific SCSR recursions that guarantee a canonical rate, that is, the fastest achievable convergence speed under generic sampling?

$Q.4$ what do the answers to $Q.1$–$Q.3$ imply for practical implementation?

The questions $Q.1$–$Q.4$ seek to understand the relationship between the errors due to recursion and sampling that naturally arise in SCSR, and their implication to SO and SRFP algorithms. As we will demonstrate through our answers, these errors are inextricably linked and fully characterizable. Furthermore, we will show that such characterization naturally leads to sampling regimes which, when combined with a deterministic recursion of a specified speed, result in specific SCSR convergence rates. The implication for implementation seems clear: given the choice of the deterministic recursive structure in use, our error characterization suggests sampling rates that should be employed in order to enjoy the best achievable SCSR convergence rates.

Remark 3.0.1. *We note that SCSR, the stochastic recursive context that we treat, is more general than (3.2). SCSR subsumes any stochastic recursion that is constructed by replacing function and derivatives appearing in a linear, superlinear, or sublinear deterministic recursion with sampled estimates. In this sense, our treatment subsumes "stochastic versions" of most popular deterministic recursions for optimization and root finding, e.g., Newton [14, 76], quasi-Newton [14, 76], fixed-point [80], trust-region, and derivative-free recursions. Specific recent examples of such stochastic recursions include [85]. We will define SCSR more rigorously in Section 3.1 where we formally present the problem statement, and in Section 3.3, where we present specific examples.*

3.1 Summary and Insight From Main Results

The results we present are broadly divided into those concerning the strong consistency of SCSR iterates, and those pertaining to SCSR's efficiency as defined from the standpoint of the total amount of simulation effort. Insight relating to consistency appears in the form of Theorem 3.4.1 and associated corollaries. Theorem 3.4.1 relates the estimator quality in SCSR with the minimum sampling rate that will guarantee almost sure convergence. Theorem 3.4.1 is generic in that it assumes little about the (deterministic) speed of the recursion in use within SCSR. A corollary of Theorem 3.4.1 is that, when using an estimator that obeys a large-deviation principle, the minimum sampling rate is logarithmic. When the estimator is of a poorer quality and has a heavy tail, the minimum sampling rate

to guarantee almost sure convergence is expectedly higher and seen to be regularly varying ("polynomial like") with index exceeding a certain threshold.

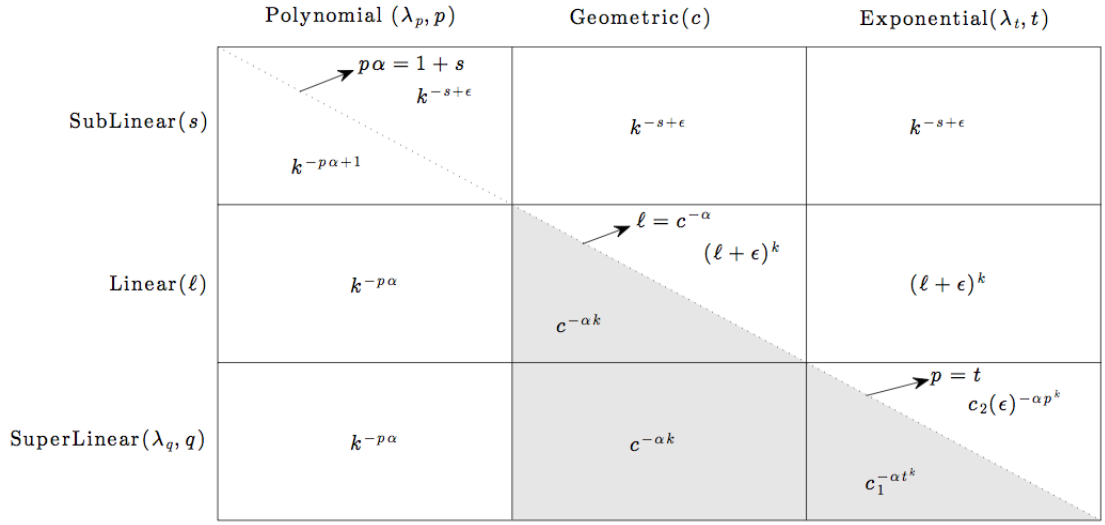

Figure 3.1: A summary of the error rates achieved by various combinations of recursion convergence rates and sampling rates. Each row corresponds to a recursion rate while each column corresponds to a sampling rate. The combinations lying below the dashed line have dominant sampling error while those above dashed line have dominant recursive error. The combinations in the shaded region are efficient in the sense that they result in the fastest possible convergence rates.

Theorems 3.5.1–3.5.5 and associated corollaries are devoted to efficiency issues surrounding SCSR. Of these, Theorems 3.5.3–3.5.5 are arguably the most important, and characterize the convergence rate of SCSR as a function of the sampling rate and the speed of recursion in use. Specifically, as summarized in Figure 3.1, these results characterize the sampling regimes resulting in predominantly sampling error ("too little sampling") versus those resulting in predominantly recursion error ("too much sampling"), along with identifying the convergence rates for all recursion-sampling combinations. Furthermore, and as illustrated using the shaded region in Figure 3.1, Theorems 3.5.3–3.5.5 identify those recursion-sampling combinations yielding the optimal rate, that is, the highest achievable convergence rates with the given simulation estimator at hand. As it turns out, and as implied by Theorems 3.5.3–3.5.5, recursions that utilize more structural information afford a wider range of sampling rates that produce the optimal rate. For instance, Theorems 3.5.3–3.5.5 imply that recursions such as (3.2) will achieve the optimal rate if the sampling rate is either geometric, or exponential up to a certain threshold; sampling rates falling outside this regime yield sub-canonical

convergence rates for SCSR. (The notions of optimal rates, sampling rates, and recursion rates will be defined rigorously in short order.) The corresponding regime when using a linearly-converging recursion such as a fixed-point recursion is narrower, and limited to a small band of geometric sampling rates. Interestingly, our results show that sublinearly converging recursions are incapable of yielding optimal rates for SCSR, that is, the sampling regime that produces optimal rates when a sublinearly converging recursion is in use is empty. We also present a result (Theorem 3.5.6) that provides a bound on the finite-time mean-squared error of SCSR iterates, under restrictive assumptions on the behavior of the recursion in use.

The rest of the chapter is organized as follows. In the ensuing section, we introduce the assumetions used throughout the chapter. This is followed by Section 3.2 where we present a rigorous problem statement, and by Section 3.3 where we present specific non-trivial examples of SCSR recursions. Sections 3.4 and 3.5 contain the main results. We provide concluding remarks in Section 3.6, with a brief commentary on implementation and the use of stochastic sample sizes.

3.2 Problem Setting and Assumptions

The general context that we consider is that of "sampling-controlled stochastic recursions," (henceforth SCSR) defined through the following recursion:

$$X_{k+1} = X_k + H(m_k, X_k), \quad k = 0, 1, 2, \ldots, \tag{SCSR}$$

where $X_k \in \mathcal{D}$ for all k, and $\mathcal{D} \subset \mathbb{R}^d$ is a known set. The "deterministic analogue" (henceforth DA) of SCSR is

$$x_{k+1} = x_k + h(x_k), \quad k = 0, 1, 2, \ldots. \tag{DA}$$

The quantity $H(m_k, X_k)$ is an element of the family of estimators $H(m, x)$ defined as

$$H(m, x) = h(x) + b(x, m) + \xi(x, m), \tag{3.3}$$

where $\mathbb{E}[\xi(x, m)] = 0$ for all $x \in \mathcal{D}$ and $m > 0$, and where $b(x, m) = \mathbb{E}[H(m, x)] - h(x)$ represents the bias of the estimator $H(m, x)$ with respect to $h(x)$. So, $H(m_k, X_k)$ appearing in SCSR should be interpreted as estimating the corresponding deterministic quantity $h(\cdot)$ at the point of interest X_k, after expending m_k amount of simulation effort. While the notation $H(m_k, \cdot)$ does not make it explicit, the estimator $H(m_k, \cdot)$ might also depend on algorithmic parameter sequences that may or may not be related to the simulation effort m_k. Such and other examples of $H(\cdot, \cdot)$ and $h(\cdot)$ are presented in the ensuing section. We also note that SCSR subsumes the Robbins-Monro [90] and Kiefer-Wolfowitz [55] processes.

3.2.1 Assumptions

The following are assumptions that will be invoked in several of the important results. Further assumptions will be made as and when required.

Assumption 3.2.1. *The set $\mathcal{D}$ is compact.*

Assumption 3.2.2. *For all $x_0 \in \mathcal{D}$, the recursion DA satisfies $\lim_{k\to\infty} x_k = x^*$, where x^* is the unique zero of $h(\cdot)$. Furthermore, we assume that such convergence is uniform in x_0, that is, for any given $\epsilon > 0$ there exists $n(\epsilon)$ (independent of x_0) such that $\|x_k - x^*\| \leq \epsilon$ for all $k \geq n(\epsilon)$ and any initial iterate x_0.*

Assumption 3.2.3. *There exists $\kappa \in \mathbb{R}$ such that for any $x, y \in \mathcal{D}$, $\|h(x) - h(y)\| \leq \kappa\|x - y\|$.*

Assumption 3.2.4. *The mean squared error of the estimator $H(m, x)$ satisfies $\sup_{x\in\mathcal{D}} \mathbb{E}[(H(m, x) - h(x))^T(H(m, x) - h(x))] = \Psi(m^{-2\alpha})$.*

The above assumptions are arguably reasonable. For instance, it is usually the case that algorithm iterates are restricted to some "large" compact set through an operation such as projection. Such practice is reflected by Assumption 3.2.1. Assumption 3.2.2 assumes convergence of the deterministic recursion DA's iterates starting from any initial point x_0. This is arguably minimal if we were to expect stochastic iterations such as SCSR to converge to the correct solution in any reasonable sense. This is because the deterministic recursion DA can be thought to be the "limiting form" of SCSR, obtained, for example, if the estimator $H(\cdot, \cdot)$ at hand is a perfect estimator of $h(\cdot)$, or through a hypothetical infinite sample. We have also assumed that the convergence in Assumption 3.2.2 happens uniformly in the initial solution x_0. The is again reasonable considering that the set $\mathcal{D}$ is compact through Assumption 3.2.1. The uniqueness of the solution x^* in Assumption 3.2.2 is purely for expositional convenience. We speculate that much of what we say can be generalized with some effort to the setting of multiple solutions using an appropriate metric such as the Hausdorff measure [87, 60].

Assumption 3.2.3 is a Lipschitz continuity type assumption on the function h that plays the role of stabilizing the iterates of DA in the vicinity of the solution. An alternative to Assumption 3.2.3 is to directly use a stability assumption such as Assumption 5.3 in [38]. Assumption 3.2.4 is a statement that the estimator $H(\cdot, \cdot)$ consistently estimates $h(\cdot)$. The constant α appearing in Assumption 3.2.4 is a measure of the quality of the estimator in use within SCSR. The assumption that the supremum (over the set $\mathcal{D}$) norm of the mean-squared error of $H(\cdot, \cdot)$ is bounded with respect to x is justified by the compactness of $\mathcal{D}$.

3.2.2 Work and Efficiency

In the analysis considered throughout this chapter, computational effort calculations are limited to simulation effort. Therefore, the total "work done" through k iterations of SCSR is given by

$$W_k = \sum_{i=1}^{k} m_i.$$

Our assessment of any sampling strategy will be based on how fast the error $E_k = \|X_k - x^*\|$ in the kth iterate of SCSR (stochastically) converges to zero *as a function of the total work* W_k. This will usually be achieved by first identifying the convergence rate of E_k with respect to the iteration number k and then translating this rate with respect to the total work W_k.

Under mild conditions, we will demonstrate that E_k cannot converge to zero faster than $W_k^{-\alpha}$ (in a certain rigorous sense), where α is defined through Assumption 3.2.4. This makes intuitive sense because it seems reasonable to expect that a stochastic recursion's quality is at most as good as the quality of the estimator at hand. We will then deem those recursions having error sequences $\{E_k\}$ that achieve the convergence rate $W_k^{-\alpha}$ as being *efficient*. The convergence rate of E_k with respect to the iteration number k is usually of little significance.

3.3 Examples

In this section, we illustrate SCSR using three popular recursions occurring within the context of SO and SRFPs. For each example, we show the specific SCSR recursion, the form of the estimator $H(\cdot, \cdot)$ and the corresponding DA recursion. We also derive the estimator convergence rate α in each case.

3.3.1 The Modified Robbins-Munro Iteration

Consider unconstrained stochastic root finding [84] on a function $g : \mathbb{R} \to \mathbb{R}$ that is estimated using $G_m(x) = m^{-1} \sum_{i=1}^{m} G_i(x)$, where $G_i(x), i = 1, 2, \ldots, m$ are iid copies of an unbiased estimator $G(x)$ of $g(x)$. (An alternative context is simulation optimization of the objective function $g : \mathbb{R} \to \mathbb{R}$ with directly observable stochastic gradient G.) In this context, the SCSR iteration is a modified form of the famous Robbins-Munro iteration [90] for stochastic root finding, obtained by dispensing with the gain sequence in the Robbins-Munro iteration and incorporating

the facility of using an iteration-dependent sample size $\{m_k\}$. Specifically:

$$X_{k+1} = X_k + \left(\hat{G}'(X_k, m_k)\right)^{-1} G_{m_k}(X_k), \tag{3.4}$$

where

$$\hat{G}'(X_k, m_k) = \frac{1}{2} s_k^{-1}(G_{m_k}(X_k + s_k) - G_{m_k}(X_k - s_k))$$

estimates the first derivative $G'(\cdot)$ at the point X_k, and s_k is the step size that is used for the computation. The recursion in (3.4) defines an SCSR recursion with $H(m, x) := \left(\hat{G}'(x, m)\right)^{-1} G_m(x)$ and $h(x) := (G'(x))^{-1} G_m(x)$. If s_k is chosen as $s_k = O_p(m_k^{-1/6})$, it can be shown under mild conditions that the error in the first-derivative estimate $\hat{G}'(x, m) - G'(x)$ is $O_p(m^{-1/3})$ [7], leading to $\mathbb{E}[(H(m, x) - h(x))^2] = \Psi(m^{-1/2})$ and hence $\alpha = 1/4$ in Assumption 3.2.4. Also, the DA recursion corresponding to (3.4),

$$x_{k+1} = x_k + (G'(x_k))^{-1} g(x_k),$$

exhibits SuperLinear(2) convergence [80].

It is worth noting here that alternative lower bias derivative estimators are possible by obtaining estimates of g at points in addition to $X_k + s_k$ and $X_k - s_k$ in the feasible region. For instance, by observing g at n design points that are strategically located, the error $\hat{G}'(x, m) - G'(x)$ can be made $O_p(m^{-n/2n+1})$, i.e., arbitrarily close to the canonical rate $O_p(m^{-1/2})$ [7].

3.3.2 The Kiefer-Wolfowitz Iteration with Naïve Hessian Estimation

Consider unconstrained simulation optimization on a function $g : \mathbb{R}^d \to \mathbb{R}$ that is estimated using $G_m(x) = m^{-1} \sum_{i=1}^{m} G_i(x)$, where $G_i(x), i = 1, 2, \ldots, m$ are iid copies of an unbiased estimator $G(x)$ of $g(x)$. Assume that the gradient of g cannot be directly observed so that the current context differs from that in Section 3.3.1. Then, the SCSR iteration is a modified Kiefer-Wolfowitz iteration [55] augmented with Hessian estimates:

$$X_{k+1} = X_k + \hat{G}^{(2)}(X_k, m_k)^{-1} \left(\frac{G_{m_k}(X_k + s_k) - G_{m_k}(X_k - s_k)}{2s_k} \right), \tag{3.5}$$

where $s_k := (s_{k1}, s_{k2}, \ldots, s_{kd})$ is the vector step used to estimate the gradient of g at $X_k := (X_{k1}, X_{k2}, \ldots, X_{kd})$ and $\hat{G}^{(2)}(x, m)$ estimates the Hessian $g^{(2)}(x)$ of

the function g at x. Assume that the (i,j)th element of the Hessian estimate $\hat{G}^{(2)}(X_k, m_k)$ is computed as usual as

$$\begin{aligned}\hat{G}^{(2)}(X_k)(i,j) &= (4s_i s_j)^{-1}(G_{m_k}(X_k + s_i e_i + s_j e_j) - G_{m_k}(X_k + s_i e_i - s_j e_j) \\ &\quad -G_{m_k}(X_k - s_i e_i + s_j e_j) + G_{m_k}(X_k - s_i e_i - s_j e_j)).\end{aligned} \tag{3.6}$$

It can be shown that if the vector step s_k is chosen as $s_{ki} = O_p(m_k^{-1/8}), i = 1, 2, \ldots, d$, then the error in the Hessian estimate $\hat{G}^{(2)}(x,m) - g^{(2)}(x)$ is $O_p(m_k^{-1/4})$, leading to $\alpha = 1/4$. (As in the context of derivative estimation described in Section 3.3.1, the error in the Hessian estimate can be made arbitrarily close to the canonical rate $O_p(m_k^{-1/2})$ by constructing more elaborate Hessian estimators based on function observations made at several strategic points. We do not go into the details here.) The DA recursion corresponding to iteration (3.5),

$$x_{k+1} = x_k + \left(g^{(2)}(x_k)\right)^{-1} \nabla g(x_k), \tag{3.7}$$

exhibits SuperLinear(2) convergence under certain structural conditions on g.

3.3.3 The Kiefer-Wolfowitz Iteration with Efficient Hessian Estimation

Consider the context of Section 3.3.2 with SCSR recursion (3.5) but when the Hessian $\hat{G}^{(2)}(X_k, m_k)$ is approximated in a more efficient manner using the simultaneous perturbation [96, 97, 98] method. Specifically, let the Hessian estimate $\hat{G}^{(2)}(x,m)$ be estimated as

$$\hat{G}^{(2)}(x,m) = \frac{1}{2}\left[\frac{\delta\hat{G}^{(1)}}{2c_k\Delta_k} + \left(\frac{\delta\hat{G}^{(1)}}{2c_k\Delta_k}\right)^T\right], \tag{3.8}$$

where $\delta\hat{G}^{(1)} = \hat{G}^{(1)}(X_k + c_k\Delta_k) - \hat{G}^{(1)}(X_k + c_k\Delta_k)$ is the difference in first-derivative estimates at points $X_k + c_k\Delta_k$ and $X_k - c_k\Delta_k$, $\hat{G}^{(1)}(x) := \dfrac{G_{m_k}(x + \tilde{c}_k\tilde{\Delta}_k) - G_{m_k}(x)}{\tilde{c}_k}$ is the first-derivative estimate at any point x, Δ_k and $\tilde{\Delta}_k$ are each vectors of independent Bernoulli random variables. (The "vector-divide" operation is followed in (3.8) as specified in [97, pp. 1841].) In [97], it is demonstrated under certain conditions that the Hessian estimate $\hat{G}^{(2)}(x,m)$ enjoys a bias $\mathcal{O}(c_k^2)$, assuming $c_k \sim \tilde{c}_k$. Under the current SCSR context, it can then be shown that the constants c_k and $\tilde{c}_k$ should be chosen as $O_p(m_k^{-1/8})$ to obtain the fastest Hessian decay rate $O_p(m_k^{-1/4})$, and correspondingly $\alpha = 1/4$. The DA recursion in this context remains (3.7) and exhibits SuperLinear(2) convergence under certain structural conditions on g.

3.4 Consistency

In this section, we present two results that clarify the conditions on the sampling rates to ensure that the iterates produced by SCSR exhibit almost sure convergence to the solution x^*. As we will see, the first of these results is generic in the sense that little is assumed about the rate of convergence of the iterates resulting from a hypothetical execution of the deterministic recursion DA.

Theorem 3.4.1. *Let the Assumptions 3.2.1 – 3.2.3 hold.*

(i) Suppose $H(m,\cdot)$ satisfies, for any $\Delta > 0$, $\sup_{x\in\mathcal{D}} Pr\{\|H(m,x) - h(x)\| > \Delta\} \to 0$ as $m \to \infty$. If the sample sizes $\{m_k\}$ are such that $m_k \to \infty$ as $k \to \infty$, then $E_k = \|X_k - x^*\| \xrightarrow{p} 0$.

(ii) Suppose $H(m,\cdot)$ satisfies, for any $\Delta > 0$, $\sup_{x\in\mathcal{D}} Pr\{\|H(m,x) - h(x)\| > \Delta\} = \mathcal{O}(r(m))$ as $m \to \infty$, where $r(m)$ is a positive-valued function that tends to zero as $m \to \infty$. If the sample sizes $\{m_k\}$ are such that $m_k \to \infty$ and $\sum_{k=1}^{\infty} r(m_k) < \infty$, then $E_k = \|X_k - x^*\| \overset{wp1}{\to} 0$.

Proof. For ease of exposition of this proof alone, we relabel SCSR and DA iterates to begin with iteration $k = 1$ instead of $k = 0$. So, SCSR iterates are $X_k, k = 1, 2, \ldots$ and DA iterates are $x_k, k = 1, 2, \ldots$ without loss in generality.

By Assumption 3.2.2, we know that for any $\delta > 0$ and any initial point $x_0 \in \mathcal{D}$, there exists $n(\delta/2) > 0$ such that the iterates $\{x_k\}$ of the recursion DA satisfy

$$\|x_k - x^*\| \leq \frac{\delta}{2}, \quad k \geq n(\delta/2). \tag{3.9}$$

(Henceforth, we drop the argument of $n(\cdot)$ for notational convenience.) Also, for illustration, divide the SCSR iterates $\{X_k\}$ into consecutive mutually exclusive and collective exhaustive groups "blocks" of size n: $(X_1, X_2, \ldots, X_n)$, $(X_{n+1}, X_{n+2}, \ldots, X_{2n})$, $(X_{2n+1}, X_{2n+2}, \ldots, X_{3n}), \ldots$. And, consider "running" an infinite number of DA sequences with initial states set equal to the "start" of each group, that is, construct the DA sequences $\{x_k^1\}, \{x_k^2\}, \ldots$ with $x_1^1 = X_1, x_1^2 = X_{n+1}, x_1^3 = X_{2n+1}, \ldots$. Due to (3.9), we see that all iterates past the initial segment of length n in each of the sequences $\{x_k^\ell\}$ is $\delta/2$ within x^*, that is,

$$\|x_k^\ell - x^*\| \leq \frac{\delta}{2}, \quad \ell = 1, 2, \ldots; k = n+1, n+2, \ldots. \tag{3.10}$$

From (3.10) and since $X_{(\ell-1)n+1} = x_1^\ell$ for $\ell = 1, 2, \ldots$ by construction, we see that if $\|X_{(\ell-1)n+j} - x_j^\ell\| \leq \delta/2$ for $\ell = 1, 2, \ldots$ and $j = 1, 2, \ldots, 2n$, then

$$\|X_{\ell n+j} - x^*\| \leq \|X_{\ell n+j} - x_{n+j}^\ell\| + \|x_{n+j}^\ell - x^*\| \leq \delta \tag{3.11}$$

for $\ell = 1, 2, \ldots$ and $j = 1, 2, \ldots n$. This implies, denoting $E_\ell := \cap_{j=1}^{2n}\{\|X_{(\ell-1)n+j} - x_j^\ell\| \leq \delta/2\}$, that

$$\cap_{\ell=1}^{\infty} E_\ell \subseteq \cap_{j=n+1}^{\infty}\{\|X_j - x^*\| \leq \delta\}. \tag{3.12}$$

To prove the assertion in (i), it is thus sufficient if we showed that when $k \to \infty$, $\Pr\{\|X_{(\ell-1)n+j} - x_j^\ell\| > \delta/2\} \to 0$ as $\ell \to \infty$ for $j = 1, 2, \ldots, n$. Likewise, (3.12) and Borel Cantelli's lemma [16] imply that the assertion in (ii) holds provided we can show that when the sample sizes $\{m_k\}$ satisfy $\sum_{k=1}^{\infty} r(m_k) < \infty$, then $\sum_{i=1}^{\infty} \Pr\{E_\ell^c\} < \infty$. Accordingly, in the rest of the proof, we will focus on the behavior of $\Pr\{\|X_{(\ell-1)n+j} - x_j^\ell\| > \delta/2\}$ and $\sum_{i=1}^{\infty} \Pr\{E_\ell^c\}$.

Recalling that the DA recursion is $x_{k+1} = x_k + h(x_k)$ for $k = 1, 2, \ldots$, we can write

$$x_{k+1} = x_1 + h(x_1) + h\left(x_1 + h(x_1)\right) + \cdots + h(\overbrace{x_1 + h(x_1) + \cdots}^{k+1 \text{ terms}}). \tag{3.13}$$

Similarly, when $X_1 = x_1$, we can write

$$X_{k+1} = x_1 + H_{m_1}(x_1) + H_{m_2}(x_1 + H_{m_1}(x_1)) + \cdots + H_{m_k}(\overbrace{x_1 + H_{m_1}(x_1) + \cdots}^{k+1 \text{ terms}}). \tag{3.14}$$

Subtracting (3.14) from (3.13), and then using Assumption 3.2.3, we get

$$\begin{aligned} \|X_{k+1} - x_{k+1}\| &\leq \sum_{j=1}^{k} \|\epsilon_j\|(1 + \kappa + \kappa^2 + \cdots + \kappa^{k-j}) \\ &\leq k \max(1, \kappa^{k-1}) \sum_{j=1}^{k} \|\epsilon_j\|, \end{aligned} \tag{3.15}$$

where $\epsilon_j = H_{m_j}(X_j) - h(X_j)$. Recalling that $X_{(\ell-1)n+1} = x_1^\ell$ for $\ell = 1, 2, \ldots$, and applying arguments leading to (3.15), we can write, for each $\ell = 1, 2, \ldots$ and each $j = 1, 2, \ldots, 2n$:

$$\begin{aligned} \|X_{(\ell-1)n+i} - x_i^\ell\| &\leq \sum_{j=(\ell-1)n+1}^{(\ell-1)n+i} \|\epsilon_j\|(1 + \kappa + \kappa^2 + \cdots + \kappa^{i-j}) \\ &\leq \beta(n, \kappa) \sum_{j=(\ell-1)n+1}^{(\ell+1)n} \|\epsilon_j\| \end{aligned} \tag{3.16}$$

where $\beta(n,\kappa) = 2n\max(1,\kappa^{2n-1})$. In (3.16), n, κ and $\beta(n,k)$ are constants. Furthermore, since $\sup_x \|H_m(x) - h(x)\| \xrightarrow{\text{p}} 0$ and $m_k \to \infty$, we have from (3.16) that $\Pr\{\|X_{(\ell-1)n+j} - x_j^\ell\| > \delta/2\} \leq \Pr\{\beta(n,k)\sum_{j=(\ell-1)n+1}^{(\ell+1)n}\|\epsilon_j\| > \delta/2\} \to 0$ as $\ell \to \infty$ for $i = 1, 2, \ldots, n$. This proves (i).

Since $E_\ell^c := \cup_{i=1}^{2n}\{\|X_{(\ell-1)n+i} - x_i^\ell\| > \delta/2\}$, using (3.16) yields

$$\begin{aligned}\Pr\{E_\ell^c\} &\leq \sum_{i=1}^{2n}\Pr\{\|X_{(\ell-1)n+i} - x_i^\ell\| > \delta/2\} \\ &\leq 2n\Pr\{\sum_{j=(\ell-1)n+1}^{(\ell+1)n}\|\epsilon_j\| > \frac{\delta}{2}\beta^{-1}(n,\kappa)\}. \end{aligned} \tag{3.17}$$

The inequality in (3.17) in turn implies that

$$\begin{aligned}\sum_{\ell=1}^{\infty}\Pr\{E_\ell^c\} &\leq 2n\sum_{\ell=1}^{\infty}\Pr\{\sum_{j=(\ell-1)n+1}^{(\ell+1)n}\|\epsilon_j\| > \frac{\delta}{2}\beta^{-1}(n,\kappa)\} \\ &\leq 2n\sum_{\ell=1}^{\infty}\sum_{j=(\ell-1)n+1}^{(\ell+1)n}\Pr\{\|\epsilon_j\| > \frac{\delta}{4n}\beta^{-1}(n,\kappa)\} \\ &\leq 4n\sum_{i=1}^{\infty}\Pr\{\|\epsilon_i\| > \frac{\delta}{4n}\beta^{-1}(n,\kappa)\} \\ &= \sum_{i=1}^{\infty}\mathcal{O}(r(m_i)) < \infty, \end{aligned} \tag{3.18}$$

where the last equality follows after noticing that n, δ, and $\beta(n,\kappa)$ are positive constants, and then applying the postulate of the theorem that $\epsilon_i = \|H(m_i, X_i) - h(X_i)\| = O_p(r(m_i))$. □

Remark 3.4.1. *The proof of Theorem 3.4.1 relies on the proof technique introduced in [38]. We deviate from [38] in that instead of assuming that the stability condition characterized through Assumption 5.3 in [38] holds, we implicitly show and use its presence. This is afforded to us through Assumption 3.2.3.*

We now state a corollary to Theorem 3.4.1 by assuming a common but specific behavior of the estimator $H(m,x)$.

Corollary 3.4.1. *Let the postulates of Theorem 3.4.1 hold. Furthermore, let the estimator error $H(m,x) - h(x)$ at $x \in \mathcal{D}$ obey a large-deviation principle with rate*

function $I_x(z)$ *that satisfies* $\inf_{x\in\mathcal{D}} \inf_{z\notin B(0,\delta)} I_x(z) > 0$ *for every* $\delta > 0$. *If the sample sizes* $\{m_k\}$ *increase at least logarithmically, that is, if* $\limsup_k m_k^{-1}(\log k)^{1+\epsilon} = 0$ *for some* $\epsilon > 0$, *then* $X_k \overset{wp1}{\to} x^*$.

Corollary 3.4.1 notes that in settings where the estimator in use obeys a large-deviation principle, strong convergence is guaranteed if the sample sizes are increased faster than logarithmically. Such settings are quite prevalent. For instance, when $H(m,\cdot)$ can be expressed as a sample mean (or an appropriate function of sample means) of independent and identically distributed random variables that have a finite moment-generating function, Cramérs theorem [31] guarantees the existence of a large-deviation principle with rate function $I_x(z)$ which is strictly convex and smooth on its effective domain, with a unique minimum attained at $z = 0$. In such cases, the condition $\inf_{x\in\mathcal{D}} \inf_{z\notin B(0,\delta)} I_x(z) > 0$ amounts to excluding pathological cases where there exists a sequence of x's in the domain $\mathcal{D}$ with rate functions that progressively become flatter. In the context of Corollary 3.4.1, all sample size growth rates considered in this chapter — Polynomial(λ_p, p), Geometric(c), and Exponential(λ_q, q) — guarantee almost sure convergence of SCSR's iterates.

Theorem 3.4.1 has been stated in terms of the function $r(m)$ to preserve generality; other corollaries based on the quality of the estimator $H(m,x) - h(x)$ can be constructed. For instance, suppose the estimator $H(m,x)$ has poorer quality than stipulated in Corollary 3.4.1 and $r(m) = \Psi(m^{-\gamma})$ for some $\gamma > 0$. Then to guarantee strong convergence, it is easily seen from Theorem 3.4.1 that the sample sizes should be increased faster than polynomial with power γ^{-1}, that is, $\{m_k\}$ should satisfy $\limsup_k m_k^{-1} k^{\gamma^{-1}+\epsilon} = 0$ for some $\epsilon > 0$.

Remark 3.4.2. *Notice that neither Theorem 3.4.1 nor Corollary 3.4.1 involves the constant* α *appearing as the index of estimator quality in Assumption 3.2.4. Part (ii) of Theorem 3.4.1 involves the tail behavior of the estimator* $H(m,x)$ *while Assumption 3.2.4 is a cruder assessment of the quality of* $H(m,x)$, *expressed using first and second moment behavior. Assumption 3.2.4 stipulates (through Chebyshev's inequality) only an upper bound on the decay rate of the tail of* $H(m,x)$.

As Corollary 3.4.1 notes, the assertive condition $\sum_{k=1}^{\infty} r(m_k) < \infty$ in part (ii) of Theorem 3.4.1 often amounts to a weak stipulation on the sample size increase rate for guaranteeing almost sure convergence. Interestingly, however, the condition $\sum_{k=1}^{\infty} r(m_k) < \infty$ is in a sense the most stringent sample size condition that guarantees almost sure convergence of SCSR iterates. This is because Theorem 3.4.1 assumes little about the rate of convergence of DA's iterates, and is hence forced to (implicitly) undertake an analysis under the slowest possible DA convergence rate.

Part (i) of Theorem 3.4.1 asserts that only the mild condition $\{m_j\} \to \infty$ on growth of the sample size sequence is needed to ensure that the iterates of SCSR converge in probability. For such a guarantee, apart from the compactness of the set $\mathcal{D}$ (Assumption 3.2.1), consistency of DA's iterates (Assumption 3.2.2), and continuity of $h(\cdot)$ (Assumption 3.2.3), only uniform consistency of the estimator $H(m, \cdot)$ is required. In other words, there is no requirement per se on the speed of convergence of the estimator $H(m, \cdot)$ to $h(\cdot)$.

3.5 Convergence Rates and Efficiency

In this section, we present results that shed light on the convergence rate and the efficiency of SCSR under different sampling and recursion contexts. Specifically, we derive the convergence rates associated with using various combinations of sample size increases (polynomial, geometric, exponential) and the speed of convergence of the DA recursion (sublinear, linear, superlinear). This information is then used to identify what sample size growth rates may be best, that is, *efficient*, for various combinations of recursive structures and simulation estimators. (See Figure 6.1 for a concise and intuitive summary of the results in this section.)

In what follows, convergence rates are first expressed as a function of the iteration k and the various constants associated with sampling and recursion. These obtained rates are then related to the total work done through k iterations of SCSR given by $W_k = \sum_{i=1}^{k} m_i$, in order to obtain a sense of the efficiency. As we will show shortly, the quantity $W_k^{-\alpha}$ is a stochastic lower bound on the error E_k in SCSR iterates; loosely speaking, $W_k^{-\alpha}$ is thus an upper bound on the *convergence rate* of the error in SCSR iterates. It is in this sense that we say SCSR's iterates are *efficient* whenever they attain the rate $W_k^{-\alpha}$.

We start with a result that provides an upper bound to SCRS's convergence rate for a given estimator quality α (see Assumption 3.2.4).

Theorem 3.5.1. *Let the Assumptions 3.2.1 – 3.2.3 hold. Also, suppose the estimator $H(m, \cdot)$ of $h(\cdot)$ satisfies:*

(i) for any $\Delta > 0$, $\sup_{x \in \mathcal{D}} Pr\{\|H(m, x) - h(x)\| > \Delta\} \to 0$ as $m \to \infty$; and

(ii) there exist $\delta, \epsilon > 0$, and $B(x^*, \epsilon')$ such that $\inf_{x \in B(x^*, \epsilon')} Pr\{m^\alpha \|H(m, x) - h(x)\| > \delta\} > \epsilon$ as $m \to \infty$.

Then the recursion SCSR cannot converge faster than $W_k^{-\alpha}$, that is, for any sequence of sample sizes $\{m_k\}$, there exist $\delta, \epsilon > 0$ such that for large enough k,

$Pr\{W_k^\alpha E_k > \delta\} > \epsilon$.

Proof. We will consider only sample size sequences $\{m_k\}$ satisfying $\{m_k\} \to \infty$. For such sequences, by Theorem 3.4.1, we are guaranteed that $\|X_k - x^*\| \xrightarrow{\text{p}} 0$.

We will show that for any sequence of sample sizes $\{m_k\} \to 0$, there exist $\delta, \epsilon > 0$ and a subsequence $\{k_j\}$ such that $\Pr\{m_{k_j}^\alpha E_{k_j} > \delta\} > \epsilon$. Since $W_k = \sum_{j=1}^k m_j \geq m_k$, the assertion of Theorem 3.5.1 will then hold.

Since $h(x)$ is continuous at x^*, we know that there exists $\delta' > 0$ such that $\|h(x)\| \leq \delta/3$ for $x \in B(x^*, \delta')$. Also, we know that since $\{X_k\} \xrightarrow{\text{p}} x^*$, $\Pr\{X_k \in B(x^*, \delta'')\} \geq 1 - \epsilon$ for large enough k, where we choose $\delta'' = \min(\delta/3, \delta', \epsilon')$. We thus know that for large enough k,

$$\begin{aligned}
&\Pr\{m_{k+1}^\alpha E_{k+1} \geq \delta/3\} \\
&\quad \geq \Pr\left\{m_{k+1}^\alpha \|X_k - x^* + H(m_k, X_k)\| \geq \delta/3 \mid X_k \in B(x^*, \delta'')\right\} \Pr\left\{X_k \in B(x^*, \delta'')\right\} \\
&\quad \geq \Pr\{m_{k+1}^\alpha \|H(m_k, X_k) - h(X_k)\| > \delta \mid X_k \in B(x^*, \delta'')\}(1 - \epsilon) \\
&\quad \geq \epsilon(1 - \epsilon) > 0, \qquad (3.19)
\end{aligned}$$

where the last inequality holds from the assumption in (ii) and since

$$X_k \in B(x^*, \delta'') \subset B(x^*, \epsilon').$$

□

Theorem 3.5.1 is important in that it provides a benchmark for efficiency. Specifically, Theorem 3.5.1 implies that sampling and recursion choices that result in errors achieving the rate $W_k^{-\alpha}$ are efficient. We emphasize that Theorem 3.5.1 only says that $W_k^{-\alpha}$ is an upper bound for the convergence rate of SCSR, and says nothing about whether this rate is in fact achievable.

The assumption in (i) of Theorem 3.5.1 is the same as that assumed in Theorem 3.4.1 and essentially states that the provided estimator $H(m, x)$ of $h(x)$ is consistent. The assumption in (ii) of Theorem 3.5.1 is a statement about the quality of the estimator $H(m, x)$; it states that the error in the estimator $H(m, x)$ does not converge faster than $m^{-\alpha}$ in an arbitrary neighborhood around the solution x^*.

We will now work towards a general lower bound on the sampling rates that achieve efficiency. We will need the following lemma for proving such a lower bound.

Lemma 3.5.1. *Let $\{a_k\}$ be any positive-valued sequence. Then*

(i) $a_k = \Psi(\sum_{j=1}^k a_j)$ if $\{a_k\}$ is faster than Geometric(c) for some $c > 1$;

(ii) $a_k = o(\sum_{j=1}^{k} a_j)$ *if* $\{a_k\}$ *is slower than Polynomial*(λ_p, p) *for some* $p > 0$.

Proof. Proof of (i). If $\{a_k\}$ increases faster than Geometric(c) for some $c > 1$, we know that $a_{k+1}/a_k \geq c > 1$ for large enough k. Hence, for some k_0 and all $k \geq j \geq k_0$, $a_j/a_k \leq c^{j-k}$. This implies that for $k \geq k_0$,

$$\begin{aligned} a_k^{-1} \sum_{j=1}^{k} a_j &\leq a_k^{-1} \sum_{j=1}^{k_0} a_j + a_k^{-1} \sum_{j=k_0+1}^{k} a_j \\ &\leq a_k^{-1} \sum_{j=1}^{k_0} a_j + \sum_{j=k_0+1}^{k} c^{j-k} \\ &= a_k^{-1} \sum_{j=1}^{k_0} a_j + \sum_{j=0}^{k-k_0-1} c^{-j} \to \frac{1}{1-c}. \end{aligned} \tag{3.20}$$

Using (3.20) and since $a_k \leq \sum_{j=1}^{k} a_j$, we conclude that the assertion holds.

Proof of (ii). Let $p > 0$ be such that $\{a_k\}$ is slower than Polyomial(λ_p, p). We then know that for some $k_0 > 0$ and all $k \geq j \geq k_0$, $a_j/a_k \geq j^p/k^p$. This implies that

$$a_k^{-1} \sum_{j=1}^{k} a_j \geq a_k^{-1} \sum_{j=1}^{k_0} a_j + a_k^{-1} \sum_{j=k_0+1}^{k} a_j \geq a_k^{-1} \sum_{j=1}^{k_0} a_j + k^{-p} \sum_{j=k_0+1}^{k} j^p. \tag{3.21}$$

Now notice that the term $k^{-p} \sum_{j=k_0+1}^{k} j^p \to \infty$ appearing on the right-hand side of (3.21) diverges as $k \to \infty$ to conclude that the assertion in (ii) holds. □

We are now ready to present a lower bound on the rate at which sample sizes should be increased in order to ensure optimal convergence rates.

Theorem 3.5.2. *The following assertions hold.*

(i) The sequence of solutions $\{X_k\}$ *satisfies* $W_k^{\alpha} E_k \xrightarrow{p} \infty$ *if* $m_{k+1} = o(\sum_{i=0}^{k} m_{i+1})$.

(ii) If $\{m_k\}$ *grows as Polynomial*(λ_p, p), *then* $W_k^{\alpha} E_k \xrightarrow{p} \infty$.

Proof. Proof of (i). We know by assumption that $W_k = \sum_{j=0}^{k} m_{j+1}$ satisfies $m_k^{\alpha} = o(W_k^{\alpha})$, implying that $W_k^{-\alpha} = o(m_k^{-\alpha})$. We also know from the proof of Theorem 3.5.1 that $m_k^{-\alpha} = \mathcal{O}_p(E_k)$. Conclude that $W_k^{-\alpha} = o(E_k)$, or alternatively $W_k^{\alpha} E_k \xrightarrow{\mathrm{p}} \infty$.

Proof of (ii). The assertion is seen to be true from (i) and upon noticing that if $\{m_k\}$ grows as Polynomial(λ_p, p), then $m_{k+1} = o(\sum_{i=0}^{k} m_{i+1})$. □

Theorem 3.5.2 makes an important assertion. It notes that for SCSR to have any chance of efficiency, sample sizes should be increased at least geometrically. This is irrespective of the speed of the recursion DA. Of course, since this is only a lower bound, increasing the sample size at least geometrically does not guarantee efficiency, which, as we shall see, depends on the speed of the DA recursion. Before we present such an efficiency result for linearly converging DA recursions, we need another lemma.

Lemma 3.5.2. *Let $\{a_j(k)\}_{j=1}^k, k \geq 1$ be a triangular array of positive-valued real numbers. Assume that the following hold.*

(i) There exists j^* and $\beta > 1$ such that $\dfrac{a_{j+1}(k)}{a_j(k)} \geq \beta$ for all $j \in [j^*, k-1]$ and all $k \geq 1$.

(ii) $\limsup_k \dfrac{a_j(k)}{a_k(k)} = \ell_j < \infty$ for each $j \in [1, j^* - 1]$.

Then

$$S_k = \sum_{i=1}^{k} a_i(k) = \mathcal{O}(a_k(k)).$$

Proof. We have, for large enough k and any $\epsilon > 0$,

$$\begin{aligned} S_k &= a_k(k) \left(\sum_{j=0}^{j^*-1} \frac{a_j(k)}{a_k(k)} + \sum_{j=j^*}^{k-1} \frac{a_j(k)}{a_k(k)} \right) \\ &\leq a_k(k) \left(j^*\epsilon + \sum_{j=0}^{j^*-1} \ell_j + \sum_{j=j^*}^{k-1} \beta^{j-k} \right), \end{aligned} \tag{3.22}$$

where the inequality follows from assumptions (i) and (ii). Since $\beta > 1, j^* < \infty$, and $\ell_j < \infty$, the term within parenthesis on the right-hand side of (3.22) is finite and the assertion holds.

□

We are now ready to prove the main result on the convergence rate and efficiency of SCSR when the DA recursion exhibits linear convergence. Theorem 3.5.3 presents the convergence rate in terms of the iteration number k first, and then in terms of the total simulation work W_k.

Theorem 3.5.3. *(Linearly Converging DA) Let Assumptions 3.2.1 – 3.2.4 hold. Suppose the deterministic recursion DA exhibits Linear(ℓ) convergence.*

Denote Geometric(c) := Ge(c). Then, recalling that $E_k := \|X_k - x^*\|$, for any $\epsilon > 0$ satisfying $\ell + \epsilon < 1$, and as $k \to \infty$:

(i)

$$E_k = \begin{cases} \mathcal{O}_p(k^{-p\alpha}), & \text{if } \{m_k\} \text{ grows as (g.a.) Polynomial}(\lambda_p, p); \\ \mathcal{O}_p\left(c^{-k\alpha}\right), & \text{if } \{m_k\} \text{ g.a. Ge}(c) \text{ with } c \in (1, \ell^{-1/\alpha}); \\ \mathcal{O}_p\left((\ell+\epsilon)^k\right), & \text{if } \{m_k\} \text{ g.a. Ge}(c) \text{ with } c \geq \ell^{-1/\alpha}; \\ \mathcal{O}_p((\ell+\epsilon)^k), & \text{if } \{m_k\} \text{ g.a. Exponential}(\lambda_t, t). \end{cases}$$

(ii)

$$\begin{array}{rcll} W_k^{\alpha\frac{p}{p+1}} E_k & = & \mathcal{O}_p(1), & \text{if } \{m_k\} \text{ g.a. Polynomial}(\lambda_p, p); \\ W_k^{\alpha} E_k & = & \mathcal{O}_p\left(1\right), & \text{if } \{m_k\} \text{ g.a. Ge}(c) \text{ with } c \in (1, \ell^{-1/\alpha}); \\ \left(c^{-\alpha}(\ell+\epsilon)^{-1}\right)^k W_k^{\alpha} E_k & = & \mathcal{O}_p\left(1\right), & \text{if } \{m_k\} \text{ g.a. Ge}(c) \text{ with } c \geq \ell^{-1/\alpha}; \\ \left(\log W_k\right)^{\log_t(1/(\ell+\epsilon))} E_k & = & \mathcal{O}_p(1), & \text{if } \{m_k\} \text{ g.a. Exponential}(\lambda_t, t). \end{array}$$

Proof. Since the postulates of Theorem 3.4.1 holds, we know that $E_k = \|X_k - x^*\| \overset{\text{wp1}}{\to} 0$. Therefore, excluding a set of measure zero, for any given $\Delta > 0$ there exists a well-defined random variable $K_0 = K_0(\Delta)$ where $\|X_k - x^*\| \leq \Delta$ for $k \geq K_0$. Now choose $\Delta = \Delta(\epsilon)$. Since $X_{k+1} = X_k + H(m_{k+1}, X_k)$ we can write

$$X_{K_0+k+1} - x^* = X_{K_0+k} - x^* + h(X_{K_0+k}) + H(m_{K_0+k+1}, X_{K_0+k}) - h(X_{K_0+k})$$

and

$$\|X_{K_0+k+1} - x^*\| \leq (\ell+\epsilon)\|X_{K_0+k} - x^*\| + \|H(m_{K_0+k+1}, X_{K_0+k}) - h(X_{K_0+k})\|. \quad (3.23)$$

Recursing (3.23) backwards and recalling the notation $E_k = \|X_k - x^*\|$, we have

$$E_{K_0+k+1} \leq (\ell+\epsilon)^k E_{K_0} + \sum_{j=K_0}^{k} (\ell+\epsilon)^{k-j} \|H(m_{K_0+j+1}, X_{K_0+j}) - h(X_{K_0+j})\|. \quad (3.24)$$

Now we notice that, even though K_0 is a random variable, $\|H(m_{K_0+k+1}, X_{K_0+k}) -$

$h(X_{K_0+k})\| = \mathcal{O}_p(m^{-\alpha}_{K_0+k+1})$ by Assumption 3.2.4. The inequality in (3.24) becomes

$$\begin{aligned} E_{K_0+k+1} &\leq (\ell+\epsilon)^k E_{K_0} + \sum_{j=K_0}^{k} (\ell+\epsilon)^{k-j} \mathcal{O}_p(m^{-\alpha}_{j+1}) \\ &= (\ell+\epsilon)^k E_{K_0} + \mathcal{O}_p\left(\sum_{j=K_0}^{k} (\ell+\epsilon)^{k-j} m^{-\alpha}_{j+1} \right). \end{aligned} \tag{3.25}$$

For ease of exposition, we drop K_0 from (3.25) and write

$$E_{k+1} \leq (\ell+\epsilon)^k E_0 + \mathcal{O}_p(\sum_{j=0}^{k} (\ell+\epsilon)^{k-j} m^{-\alpha}_{j+1}). \tag{3.26}$$

(There is no loss in generality between (3.25) and (3.26) because we are interested only in the tail behavior of the sequence $\{E_k\}$ and neither the starting error $E_{K_0} \leq \text{diam}(\mathcal{D}) < \infty$ nor the starting sample m_{K_0+1} are of relevance to the analysis that follows.)

We will now show that the first equality in assertion (i) of Theorem 3.5.3 holds by showing that the two assumptions of Lemma 3.5.2 hold for $\sum_{j=0}^{k} (\ell+\epsilon)^{k-j} m^{-\alpha}_{j+1}$ appearing in (3.26). Set the summand of $\sum_{j=0}^{k} (\ell+\epsilon)^{k-j} m^{-\alpha}_{j+1}$ to $a_j(k)$ and since $m_j = m_0 j^p$, we have $\dfrac{a_{j+1}(k)}{a_j(k)} = \dfrac{1}{(\ell+\epsilon)} (\dfrac{j+1}{j+2})^{p\alpha}$. Choosing β such that $\beta > 1$ and $(\ell+\epsilon)\beta < 1$, and setting $j^* = \text{Max}(1, \dfrac{1}{1-((\ell+\epsilon)\beta)^{1/\alpha}} - 2)$, we see that the first assumption of Lemma 3.5.2 is satisfied. The second assumption of Lemma 3.5.2 is also satisfied since for any fixed $j^* > 0$, $\limsup_k \dfrac{a_j}{a_k} = \limsup_k (\ell+\epsilon)^{k-j} (\dfrac{k+1}{j+1})^{p\alpha} = 0$ for all $j \in [1, j^*]$.

To prove the second and third equalities in assertion (i) of Theorem 3.5.3, suppose $\{m_k\}$ grows as Geometric(c) with $c < (\ell+\epsilon)^{-1/\alpha}$, that is, $c^{-\alpha} > \ell + \epsilon$. Then, noticing that $m_j = m_0 c^j$, we write

$$\begin{aligned} \sum_{j=0}^{k} (\ell+\epsilon)^{k-j} m^{-\alpha}_{j+1} = m_0^{-\alpha} \sum_{j=0}^{k} (\ell+\epsilon)^{k-j} c^{-(j+1)\alpha} &= m_0^{-\alpha} c^{-\alpha(k+1)} \frac{(1 - (\dfrac{\ell+\epsilon}{c^{-\alpha}})^{k+1})}{1 - \dfrac{\ell+\epsilon}{c^{-\alpha}}} \\ &= \Psi(c^{-\alpha(k+1)}), \end{aligned} \tag{3.27}$$

and use (3.27) in (3.26). If $\{m_k\}$ grows as Geometric(c) with $c > (\ell+\epsilon)^{-1/\alpha}$, that

is, $c^{-\alpha} < \ell + \epsilon$, then notice that (3.27) becomes

$$\begin{aligned}\sum_{j=0}^{k}(\ell+\epsilon)^{k-j}m_{j+1}^{-\alpha} &= m_0^{-\alpha}\sum_{j=0}^{k}(\ell+\epsilon)^{k-j}c^{-(j+1)\alpha} \\ &= m_0^{-\alpha}(\frac{c^{-\alpha}}{\ell+\epsilon})(\ell+\epsilon)^{k+1}\frac{(1-(\frac{c^{-\alpha}}{\ell+\epsilon})^{k+1})}{1-\frac{c^{-\alpha}}{\ell+\epsilon}} \\ &= \Psi((\ell+\epsilon)^{k+1}). \end{aligned} \tag{3.28}$$

Now use (3.28) in (3.26).

To see that the fourth equality in assertion (i) of Theorem 3.5.3 holds, we notice that a sample size sequence $\{m_k\}$ that grows as Exponential(λ_t, t) is faster than a sample size sequence $\{m_k\}$ that grows as Geometric(c) for any $c, t, \lambda_t \in (0, \infty)$. *Proof of (ii).* To prove the assertion in (ii), we notice that since $W_k = \sum_{j=1}^{k} m_j$, and we have

$$\begin{aligned} W_k &= \Psi(k^{p+1}) \text{ if } \{m_k\} \text{ grows as Polynomial}(\lambda_p, p); \\ &= \Psi(c^k) \text{ if } \{m_k\} \text{ grows as Geometric}(c); \\ &= \Psi\left((\lambda_t^{\frac{1}{t-1}}m_0)^{t^k}\right) \text{ if } \{m_k\} \text{ grows as Exponential}(\lambda_t, t). \end{aligned} \tag{3.29}$$

Now use (3.29) in assertion (i) to obtain the assertion in (ii). □

Theorem 3.5.3 provides various insights about the behavior of the error in SCSR iterates. For instance, the error structures detailed in (i) of Theorem 3.5.3 suggest two well-defined sampling regimes where only one of the two error types, sampling error or recursion error, is dominant. Specifically, note that $E_k = \mathcal{O}_p(k^{-p\alpha})$ when the sampling rate is Polyonimal(λ_p, p). This implies that when DA exhibits Linear(ℓ) convergence, polynomial sampling is "too little" in the sense that SCSR's convergence rate is dictated purely by sampling error (since the constant c corresponding to DA's convergence is absent in the expression for E_k). The corresponding reduction in efficiency can be seen in (ii) where E_k is shown to converge as $\mathcal{O}_p(W_k^{-\alpha\frac{p}{1+p}})$. (Recall that efficiency amounts to $\{E_k\}$ achieving a convergence rate $\mathcal{O}_p(W_k^{-\alpha})$.)

The case that is diametrically opposite to polynomial sampling is exponential sampling, where the sampling is "too much" in the sense that the convergence rate $E_k = \mathcal{O}_p((\ell+\epsilon)^k)$ is dominated by recursion error. There is a corresponding reduction in efficiency as can be seen in the expression provided in (ii) of Theorem 3.5.3.

The assertion (ii) in Theorem 3.5.3 also implies that the only sampling regime that achieves efficiency for linearly converging DA recursions is a Geometric(c) sampling rate with $c \in (1, \ell^{-1/\alpha})$. Values of c on or above the threshold $\ell^{-1/\alpha}$ result in "too much" sampling in the sense of a dominating recursion error and a corresponding reduction in efficiency, as quantified in (i) and (ii) of Theorem 3.5.3.

We now state a result that is analogous to Theorem 3.5.3 for the context of super-linearly converging DA recursions.

Theorem 3.5.4. *(SuperLinearly Converging DA) Let Assumptions 3.2.1 – 3.2.4 hold. Suppose the deterministic recursion DA exhibits SuperLinear(λ_q, q) convergence. Then, for any $\epsilon > 0$ and as $k \to \infty$:*

(i)

$$E_k = \begin{cases} \mathcal{O}_p(k^{-p\alpha}), & \text{if } \{m_k\} \text{ grows as Polynomial}(\lambda_p, p); \\ \mathcal{O}_p\left(c^{-\alpha k}\right), & \text{if } \{m_k\} \text{ grows as Geometric}(c); \\ \mathcal{O}_p(c_1^{-\alpha t^k}), & \text{if } \{m_k\} \text{ grows as Exponential}(\lambda_t, t), t < q; \\ \mathcal{O}_p(c_2(\epsilon)^{-\alpha q^k}), & \text{if } \{m_k\} \text{ grows as Exponential}(\lambda_t, t), t \geq q; \end{cases}$$

(ii)

$$\begin{aligned} W_k^{\alpha\frac{p}{p+1}} E_k &= \mathcal{O}_p(1), && \text{if } \{m_k\} \text{ grows as Polynomial}(\lambda_p, p); \\ W_k^{\alpha} E_k &= \mathcal{O}_p\left(1\right), && \text{if } \{m_k\} \text{ grows as Geometric}(c); \\ W_k^{\alpha} E_k &= \mathcal{O}_p(1), && \text{if } \{m_k\} \text{ grows as Exponential}(\lambda_t, t), t < q; \\ c_2^{\alpha p^{\log_t \log_{c_1} W_k}} E_k &= \mathcal{O}_p(1), && \text{if } \{m_k\} \text{ grows as Exponential}(\lambda_t, t), t \geq q; \end{aligned}$$

where $c_1 = m_0 \lambda_t^{1/(t-1)}$ and $c_2(\epsilon) = m_0^{t/q}(\lambda_q + \epsilon)^{1/q} 2^{-1/\alpha q}$.

Proof. Repeating arguments leading to (3.23) in the proof of Theorem 3.5.3), we write

$$E_{K_0+k+1} \leq (\lambda_q + \epsilon) E_{K_0+k}^q + \|\zeta_{K_0+k}\| \tag{3.30}$$

where $\zeta_{K_0+k} = \|h(X_{K_0+k}) - H(m_{K_0+k+1}, X_{K_0+k})\|$, $K_0 = K_0(\Delta)$ satisfies $\|X_k -$

$x^*\| \leq \Delta$ except for a set of measure zero. We then recurse (3.30) to obtain

$$\begin{aligned} E_{K_0+k+1} &\leq (\lambda_q+\epsilon)E_{K_0+k}^q + \|\zeta_{K_0+k}\| \\ &\leq (\lambda_q+\epsilon)\left((\lambda_q+\epsilon)E_{K_0+k-1}^q + \|\zeta_{K_0+k-1}\|\right)^q + \|\zeta_{K_0+k}\| \\ &\leq (\lambda_q+\epsilon)\left(2^q(\lambda_q+\epsilon)^q E_{K_0+k-1}^{q^2} + 2^q(\lambda_q+\epsilon)\|\zeta_{K_0+k-1}\|^q\right) + \|\zeta_{K_0+k}\| \\ &= 2^q(\lambda_q+\epsilon)^{1+q}E_{K_0+k-1}^{q^2} + 2^q(\lambda_q+\epsilon)\|\zeta_{K_0+k-1}\|^q + \|\zeta_{K_0+k}\| \\ &\vdots \\ &\leq 2^{s(k+1)-1}(\lambda_q+\epsilon)^{s(k+1)}E_{K_0}^{q^{k+1}} + \sum_{j=K_0}^{K_0+k}\|\zeta_j\|^{q^{k-j}}(\lambda_q+\epsilon)^{s(k-j)}2^{s(k-j+1)-1} \end{aligned} \tag{3.31}$$

where $s(j) = 1+q+q^2+\cdots+q^{j-1}$. As in the proof of Theorem 3.5.3, we notice that $\zeta_j = \mathcal{O}_p(m_{j+1}^{-\alpha})$ and also drop K_0 from (3.31) for ease of exposition to get

$$\begin{aligned} E_{k+1} &\leq 2^{s(k+1)-1}(\lambda_q+\epsilon)^{s(k+1)}E_0^{q^{k+1}} + \mathcal{O}_p\left(\sum_{j=0}^{k} m_{j+1}^{-\alpha q^{k-j}}(\lambda_q+\epsilon)^{s(k-j)}2^{s(k-j+1)-1}\right) \\ &\leq 2^{\rho(q)q^k}(\lambda_q+\epsilon)^{\rho(q)q^k}E_0^{q^{k+1}} + \mathcal{O}_p\left(\sum_{j=0}^{k}((\lambda_q+\epsilon)^{1/q}2)^{\rho(q)q^{k-j}}m_{j+1}^{-\alpha q^{k-j}}\right), \end{aligned} \tag{3.32}$$

where $\rho(q) = q/(q-1)$.

Proof of (i). We will now prove the first three equalities of (i) hold by showing that the two postulates of Lemma 3.5.2 hold for $\sum_{j=0}^{k}((\lambda_q+\epsilon)^{1/q}2)^{\rho(q)q^{k-j}}m_{j+1}^{-\alpha q^{k-j}}$ appearing in (3.32), thereby proving that $\sum_{j=0}^{k}((\lambda_q+\epsilon)^{1/q}2)^{\rho(q)q^{k-j}}m_{j+1}^{-\alpha q^{k-j}}$ is of the same order as the last summand $((\lambda_q+\epsilon)^{1/q}2)m_{k+1}^{-\alpha}$. Towards this, set $a_j(k) = ((\lambda_q+\epsilon)^{1/q}2)^{\rho(q)q^{k-j}}m_{j+1}^{-\alpha q^{k-j}}$ and we have

$$\frac{a_{j+1}(k)}{a_j(k)} = \left(\frac{m_{j+1}^q}{(2(\lambda_q+\epsilon)^{1/q})^{\rho(q)\frac{q-1}{\alpha}}m_{j+2}}\right)^{\alpha q^{k-j-1}}.$$

If $\{m_j\}$ grows as Polynomial(λ_p, p), Geometric(c), or Exponential(λ_t, t) with $t < q$, some algebra yields that $a_{j+1}(k)/a_j(k) > \beta$ for any $\beta \in (0,\infty)$ and large-enough j. Thus, the first postulate of Lemma 3.5.2 is satisfied when $\{m_j\}$ is Polynomial(λ_p, p), Geometric(c), or Exponential(λ_t, t) with $t < q$.

Also, since

$$\frac{a_j}{a_k} = (2(\lambda_q+\epsilon)^{1/q})^{(\rho(q)-1)q^{k-j}}\left(\frac{m_{k+1}}{m_{j+1}^{q^{k-j}}}\right)^{\alpha},$$

some algebra again yields that $\limsup_k a_j(k)/a_k(k) \to 0$ for j lying in any fixed interval for the three cases Polynomial(λ_p, p), Geometric(c), and Exponential(λ_t, t) with $t < q$, and hence the second postulate of Lemma 3.5.2 is satisfied as well. We thus conclude that the first three equalities in (i) hold.

To obtain the last equality in (i), write $\sum_{j=0}^{k}((\lambda_q+\epsilon)^{1/q}2)^{\rho(q)q^{k-j}} m_{j+1}^{-\alpha q^{k-j}}$ appearing in (3.32) as

$$\sum_{j=0}^{k}((\lambda_q+\epsilon)^{1/q}2)^{q^{k-j}} m_{j+1}^{-\alpha q^{k-j}} = \sum_{j=0}^{k}((\lambda_q+\epsilon)^{1/q}2)^{\rho(q)q^{j}} m_{k-j+1}^{-\alpha q^{j}}.$$

Again some algebra yields that if $\{m_j\}$ grows as Exponential(λ_t, t) with $t \geq q$, $a_{j+1}(k)/a_j(k) > \beta$ for large enough j. Thus, the first postulate of Lemma 3.5.2 is satisfied when $\{m_j\}$ is Exponential(λ_t, t) with $t \geq q$. Also, since

$$\frac{a_j(k)}{a_k(k)} = (2(\lambda_q+\epsilon)^{1/q})^{\rho(q)(q^j-q^k)}(\frac{m_1^{q^k}}{m_{k-j+1}^{q^j}})^{\alpha},$$

some algebra again yields that $\limsup_k a_j(k)/a_k(k) \to 0$ for j lying in any fixed interval when $\{m_k\}$ is Exponential(λ_t, t) with $t \geq q$, and hence the second postulate of Lemma 3.5.2 is satisfied as well. We thus conclude that the last equality in (i) holds.

Proof of (ii). To prove the assertion in (ii), we notice that since $W_k = \sum_{j=1}^{k} m_j$, and the expressions in (3.29) hold here as well. Now use (3.29) in assertion (i) to obtain the assertion in (ii).

□

Theorem 3.5.4 is the analogue to Theorem 3.5.3 but for superlinearly converging DA recursions. Like Theorem 3.5.3, Theorem 3.5.4 demonstrates that there are two well-defined sampling regimes corresponding to predominant sampling and recursion errors. Assertion (i) in Theorem 3.5.4 implies that all of polynomial sampling, all of geometric sampling, and part of exponential sampling result in predominant sampling error. This makes intuitive sense because the rapidly decaying recursion error demands (or allows) additional sampling to drive the sampling error down at the same rate as the recursion error. Correspondingly, as (ii) in Theorem 3.5.4 implies, there is a larger range of sampling rates that result in efficiency; in some sense, this is the advantage of using a faster DA recursion. Specifically, (ii) in Theorem 3.5.4 implies that any Geometric(c) ($c \in (0, \infty)$) sampling is efficient, that is, achieves the $W_k^{-\alpha}$ rate, when the DA recursion is superlinear. And, Exponential(λ_t, t) sampling results in efficiency as long as $t < q$.

We next prove a result analogous to Theorems 3.5.3 and 3.5.4 but for the case where the DA recursion exhibits SubLinear(s) convergence.

Theorem 3.5.5. *(SubLinearly Converging DA) Let Assumptions 3.2.1 – 3.2.4 hold. Suppose the deterministic recursion DA exhibits SubLinear(λ_s, s) convergence. Then, for any $\epsilon > 0$ satisfying $\epsilon < s$, and as $k \to \infty$:*
(i)

$$E_k = \begin{cases} \mathcal{O}_p(k^{-p\alpha+1}), & \text{if } \{m_k\} \text{ grows as Polynomial}(\lambda_p, p), p\alpha < s+1; \\ \mathcal{O}_p(k^{-s+\epsilon}), & \text{if } \{m_k\} \text{ grows as Polynomial}(\lambda_p, p), p\alpha \geq s+1; \\ \mathcal{O}_p(k^{-s+\epsilon}), & \text{if } \{m_k\} \text{ grows as Geometric}(c); \\ \mathcal{O}_p(k^{-s+\epsilon}), & \text{if } \{m_k\} \text{ grows as Exponential}(\lambda_t, t); \end{cases}$$

(ii)

$$\begin{aligned} W_k^{\alpha\frac{1-1/(p\alpha)}{1+1/p}} E_k &= \mathcal{O}_p(1), && \text{if } \{m_k\} \text{ grows as Polynomial}(\lambda_p, p), p\alpha < s+1; \\ W_k^{\alpha\frac{s-\epsilon}{p\alpha+\alpha}} E_k &= \mathcal{O}_p(1), && \text{if } \{m_k\} \text{ grows as Polynomial}(\lambda_p, p), p\alpha \geq s+1; \\ (\log_c W_k)^{s+\epsilon} E_k &= \mathcal{O}_p(1), && \text{if } \{m_k\} \text{ grows as Geometric}(c); \\ \log_t(\log_{c_1} W_k) E_k &= \mathcal{O}_p(1), && \text{if } \{m_k\} \text{ grows as Exponential}(\lambda_t, t); \end{aligned}$$

where $c_1 = m_0 \lambda_t^{1/(t-1)}$.

Proof. Repeating arguments leading to (3.23) in the proof of Theorem 3.5.3, we write

$$\begin{aligned} E_{K_0+k+1} &\leq \|X_{K_0+k} - x^* + h(X_{K_0+k})\| + \|\zeta_{K_0+k}\| \\ &\leq E_k(1 - \frac{s-\epsilon}{K_0+k}) + \|\zeta_k\| \end{aligned} \tag{3.33}$$

where $\zeta_{K_0+k} = \|h(X_{K_0+k}) - H(m_{K_0+k+1}, X_{K_0+k})\|$, $K_0 = K_0(\Delta)$ satisfies $\|X_k - x^*\| \leq \Delta$ except for a set of measure zero, We then recurse (3.33) to obtain

$$\begin{aligned} E_{K_0+k} &\leq E_{k_0}\left(\Pi_{j=K_0+1}^{k}(1 - \frac{s-\epsilon}{j})\right) + \sum_{j=K_0}^{k} \zeta_j \Pi_{i=j+1}^{k}(1 - \frac{s-\epsilon}{i}) \\ &= E_{K_0}\left(\Pi_{j=K_0+1}^{k}(1 - \frac{s-\epsilon}{j})\right) + \mathcal{O}_p(\sum_{j=K_0}^{k} m_{j+1}^{-\alpha} \Pi_{i=j+1}^{k}(1 - \frac{s-\epsilon}{i})) \\ &= \Psi(k^{-s+\epsilon}) + k^{-s+\epsilon}\mathcal{O}_p(\sum_{j=K_0}^{k} m_{j+1}^{-\alpha}(j+1)^{s-\epsilon}), \end{aligned} \tag{3.34}$$

where the last equality is evident upon noticing that $\Pi_{i=j+1}^{k}(1-\frac{s-\epsilon}{i})) = \Psi((\frac{j+1}{k})^{s-\epsilon})$.

If $\{m_k\}$ grows as Polynomial(λ_p, p), as $k \to \infty$,

$$\sum_{j=K_0}^{k} m_{j+1}^{-\alpha}(j+1)^{s-\epsilon} = \sum_{j=K_0}^{k} \lambda_p^{-\alpha}(j+1)^{-p\alpha+s-\epsilon} = \Psi(k^{-p\alpha+s-\epsilon+1}). \tag{3.35}$$

If $\{m_k\}$ grows as Geometric(c), as $k \to \infty$,

$$\sum_{j=K_0}^{k} m_{j+1}^{-\alpha}(j+1)^{s-\epsilon} = \sum_{j=K_0}^{k} m_0 c^{-(j+1)\alpha}(j+1)^{s-\epsilon} = \Psi(1). \tag{3.36}$$

If $\{m_k\}$ grows as Exponential(λ_t, t), as $k \to \infty$,

$$\sum_{j=K_0}^{k} m_{j+1}^{-\alpha}(j+1)^{s-\epsilon} = \sum_{j=K_0}^{k} m_0^{-\alpha t^{k+1}} \lambda_t^{\alpha/(t-1)} \lambda_t^{-\alpha t^{k+1}/(t-1)}(j+1)^{s-\epsilon} = \Psi(1). \tag{3.37}$$

Now use (3.35), (3.36), and (3.37) in (3.33) to see that the assertion in (i) holds.

Proof of (ii). To prove the assertion in (ii), we again notice that since $W_{k+1} = \sum_{j=1}^{k} m_k$, and the expressions in (3.29). Now use (3.29) in assertion (i) to obtain the assertion in (ii). □

The context of Theorem 3.5.5, sublinearly converging DA recursions, is diametrically opposite to that of Theorem 3.5.4 in the sense that most sampling regimes result in predominantly recursion error. Specifically, (i) in Theorem 3.5.5 implies that all geometric, all exponential, and a portion ($p\alpha \geq s+1$) of polynomial sampling result in dominant recursion error. Perhaps more importantly, (ii) in Theorem 3.5.5 implies that there is no sampling regime that results in efficiency when the DA recursion exhibits sublinear convergence. The best achievable rate under these conditions is $W_k^{-\alpha\eta^*}$ where $\eta^* = (p\alpha - 1)/(p\alpha + \alpha) \in (0, 1)$, obtained through Polynomial(λ_p, p) sampling as $p \to (s+1)\alpha^{-1}$.

We will end this section with a finite-time bound on the mean-squared error of SCSR's iterates assuming that the DA recursion exhibits a linear contraction at every step, and not just asymptotically. This context is surprisingly frequent and occurs, for example, when optimizing a strongly convex function using a linearly converging DA recursion. The utility of Theorem 3.5.6 is clear — to identify the number of steps of the algorithm required to achieve a mean squared error below a specified threshold. Analogous cruder bounds for stochastic approximation have appeared recently [75].

Theorem 3.5.6. *(Finite-time bounds.) Let the deterministic recursion DA be such that*

$$\|X_{k+1}^{(i)} - x^*\| \leq \ell \|X_k^{(i)} - x^*\|, \quad k = 0, 1, \ldots \tag{3.38}$$

for all $i \in \{1, 2, \ldots, d\}$, some $\ell \in (0, 1)$, and recalling the notation

$$X_k := (X_k^{(1)}, X_k^{(2)}, \ldots, X_k^{(d)}).$$

Also suppose that there exists $\sigma < \infty$ such that $\sup_{x \in \mathcal{D}} \mathbb{E}[(H^{(i)}(m, x) - h^{(i)}(x))^2] \leq \sigma^2 m^{-2\alpha}$ for all $i \in \{1, 2, \ldots, d\}$, where α is defined through Assumption 3.2.4. Then

$$\begin{aligned} \mathrm{mse}(\|X_{k+1} - x^*\|) &:= \mathbb{E}[\|X_{k+1} - x^*\|^2] \\ &\leq 2d\sigma^2 \left(\sum_{j=0}^{k} m_{j+1}^{-2\alpha} \right) + d \left(\ell^{k+1} \|X_0 - x^*\| + \sigma \sum_{j=0}^{k} \ell^{k-j} m_{j+1}^{-\alpha} \right)^2. \end{aligned}$$

Proof. Using (3.38) we write for $i \in \{1, 2, \ldots, d\}$ that

$$\begin{aligned} \mathbb{E}[|X_{k+1}^{(i)} - x^{*(i)}|] &= \mathbb{E}[|X_k^{(i)} - x^{*(i)} + h^{(i)}(X_k^{(i)})] + H^{(i)}(X_k^{(i)}, m_{k+1}) - h^{(i)}(X_k^{(i)})|] \\ &\leq \ell \, \mathbb{E}[|X_k^{(i)} - x^{*(i)}|] + \frac{\sigma}{m_{k+1}^{\alpha}} \\ &\leq \ell^2 \mathbb{E}[|X_{k-1}^{(i)} - x^{*(i)}|] + \ell \frac{\sigma}{m_k^{\alpha}} + \frac{\sigma}{m_{k+1}^{\alpha}} \\ &\vdots \\ &= \ell^{k+1} \|X_0 - x^*\| + \sigma \sum_{j=0}^{k} \ell^{k-j} m_{j+1}^{-\alpha}. \end{aligned} \tag{3.39}$$

Next, we write

$$\begin{aligned} \mathrm{Var}(X_{k+1}^{(i)}) &= \mathrm{Var}(X_k^{(i)}) + \mathrm{Var}(H^{(i)}(X_k, m_{k+1})) + 2\mathrm{Cov}(X_k^{(i)}, H^{(i)}(X_k, m_{k+1})) \\ &\leq 2\mathrm{Var}(X_k^{(i)}) + 2\mathrm{Var}(H(m_{k+1}, X_k^{(i)})). \end{aligned} \tag{3.40}$$

Recurse (3.40) to obtain

$$\mathrm{Var}(X_{k+1}^{(i)}) \leq 2\sigma^2 \sum_{j=0}^{k} m_{j+1}^{-2\alpha}, i \in \{1, 2, \ldots, d\}. \tag{3.41}$$

Using (3.39) and (3.41) in

$$\begin{aligned} \mathrm{mse}(\|X_{k+1} - x^*\|) &:= \mathbb{E}[\|X_{k+1} - x^*\|^2] \\ &= \sum_{i=1}^{d} \mathrm{Var}(X_{k+1}^{(i)}) + \mathbb{E}^2[|X_{k+1}^{(i)} - x^{*(i)}|], \end{aligned}$$

the result follows.

□

Results analogous to Theorem 3.5.6 but for other types of DA recursions should be obtainable in a similar fashion. We emphasize that Theorem 3.5.6 makes no assumption about the compactness of $\mathcal{D}$ because a linear contraction is assumed to be in effect in every step of the DA recursion. In Corollary 3.5.1 below, we specialize Theorem 3.5.6 for the case where the sample sizes increase as Geometric(c). Analogous results for other sampling rates should be similarly obtainable.

Corollary 3.5.1. *(Finite-time Bound for Geometric(c) Sample Sizes.) Suppose that the postulates of Theorem 3.5.6 hold. Also, let the sample size sequence* $\{m_k\}$ *increase as Geometric(c). Then, in the efficient regime, that is, when* $\ell c^\alpha < 1$*, the mean squared error of the solution* X_{k+1} *satisfies*

$$\mathbb{E}[\|X_{k+1} - x^*\|^2] \leq d\left(\ell^{k+1}\|X_0 - x^*\| + \frac{\sigma}{m_0^\alpha} c^{-\alpha(k+1)}\left(\frac{1-(\ell c^\alpha)^{k+1}}{1-\ell c^\alpha}\right)\right)^2 + 2d\sigma^2 m_0^{-2\alpha}\frac{1-c^{-2\alpha(k+1)}}{1-c^{-2\alpha}}.$$

Moreover, the bound appearing in (3.42) is strictly decreasing for $c \in (0, \ell^{-1/\alpha})$.

Proof. Use $m_j = c^j m_0$ in Theorem 3.5.6.

□

3.6 Concluding Remarks

The use of simulation-based estimators within well-established algorithmic recursions is becoming an attractive paradigm to solve optimization and root-finding problems in contexts where the underlying functions can only be estimated. In such contexts, the question of how much to simulate (towards estimating function and derivative values at a point) becomes important particularly when the available simulations are computationally expensive. In this chapter, we have argued that there is an interplay between the structural error inherent in the recursion in use and the sampling error inherent in the simulation estimator. Our characterization of this interplay provides guidance (see Figure 3.1) on how much sampling should be undertaken under various recursive contexts in order to ensure that the resulting iterates are provably efficient.

A few other comments relating to the results we have presented and about ongoing research are now in order.

(i) We have assumed throughout that the solution to the deterministic recursion DA is unique. This, of course, is not always the case; in contexts where there are multiple solutions, we believe that our sampling recommendations will continue to hold after altering the way we measure the accuracy of SCSR's iterates. For example, while we have used the L^2 norm $\|X_k - x^*\|$ as a measure of accuracy, the context of multiple solutions will demand the use of alternative measures such as the Hausdorff distance [87] between X_k and the set of zeros of the function h, or the L^2 norm $\|h(X_k)\|$ in the function space.

(ii) None of the results we have presented are of the "central limit theorem" type; they are cruder and of the $O_p(\cdot)$ type. This is because, when the sampling and recursion choices lie off the diagonal in Figure 3.1, either the recursion error or the sampling are dominant and consequently lead to a situation where the contribution to the error in the SCSR iterates is due only to a few terms. When the sampling and recursion choices lie on the diagonal, a central-limit theorem will likely hold but characterizing such a fine result will involve further detailed assumptions on the convergence characteristics of the deterministic recursion.

(iii) Another interesting question that we have not treated here is that of iterate averaging [88] to increase efficiency. Recall that our results suggest that efficiency cannot be achieved for sampling regimes slower than geometric. it is possible that iterate averaging might be useful in such sub-geometric low-sampling regimes, e.g., polynomial. It also seems that such averaging is of less value in high sampling regimes for the same reason that CLT-type results do not take hold due to the Lindberg-Feller condition [93] failing on constituent sums.

(iv) All of the results we have presented assume that the sequence of sample sizes $\{m_k\}$ used within SCSR is deterministic. To this extent, our results provide guidance on only the *rate* at which sampling should be performed in order that SCSR's iterates remain efficient. We envision that an implementable algorithm will dynamically choose sample sizes as a function of the observed trajectory of the algorithm while ensuring that the increase rates prescribed by our results are followed. Accordingly the main focus in the next chapter is to investigate such particular strategies within the context of SCSR.

Chapter 4

Adaptive-SCSR

Chapter 3 explored the notion of optimal sample size regimes for stochastic recursions such as (3.2), by introducing and analyzing the broader context of Sampling-Controlled Stochastic Recursions (SCSR) for finding the zero of an unknown function $h : \mathbb{R}^d \to \mathbb{R}^d$. In this chapter we let the main estimator defined within (SCSR) be simply proportional to the estimator of h. Letting

$$H(m, x) = -\frac{1}{\beta}\tilde{H}(m, x),$$

where $\tilde{H}(m, x)$ is a consistent estimator of h at x, and β is a positive constant, the stochastic recursion considered here is as follows:

$$X_{k+1} = X_k - \frac{1}{\beta}\tilde{H}(M_k, X_k). \tag{SCSR}$$

Chapter 3 characterized the rates at which the sample sizes $\{M_k\}$ should increase in (SCSR) in order to guarantee the consistency and efficiency of the resulting iterates. In particular, we demonstrated that the speed of the underlying recursive function h, its estimator $\tilde{H}(M_k, X_k)$, and the optimal regime of sample sizes are intimately linked, with faster recursions allowing for a wider range of sample sizes while remaining efficient. For instance, Chapter 3 showed that when linearly converging recursions are employed, certain geometrically increasing sample size sequences are efficient; likewise, when superlinearly converging recursions are employed, all geometrically increasing sample size sequences, and certain super-exponential sample-size sequences are efficient.

The sampling regimes characterized in Chapter 3 constituted an important step towards making stochastic recursions implementable, since they serve to provide some guidance on sampling. Such guidance is only broad, however, and still leaves

a lot of room for choice. For instance, when using linearly converging recursions, Chapter 3 demonstrated that choosing $\{M_k\}$ such that $M_k/M_{k-1} = \gamma$, for all $k \geq 2$ produces iterates that are consistent and efficient (in a certain rigorous sense) as long as $\gamma \in (0, c)$, where $c > 1$ is a constant. This directive is useful but, depending on the specific problem, different choices of γ, or even varying γ across iterations, may be needed to produce robust algorithm performance. In general, good algorithm performance in finite time entails inferring and reacting to specific problem structure, perhaps by using the trajectory of algorithm iterates and their corresponding function estimates. Regardless of the problem structure, the analysis in Chapter 3 is asymptotic, leaving an enormous range of possible choices of $\{M_k\}$ that still guarantee efficiency.

Can sample sizes $\{M_k\}$ be chosen adaptively, by reacting to function information that is obtained as the iterates evolve through the search space? Moreover, can such adaption happen in way that also ensures consistency and efficiency in the rigorous sense of Chapter 3? There have been some recent proposals in the literature towards answering this question. For instance [25] propose the following two-stage sampling procedure to determine the sample size at any iteration k:

$$M_k = \frac{\alpha \hat{\sigma}^2(M_{k-1}, X_k)}{\|\tilde{H}(M_{k-1}, X_k)\|^2}, \tag{4.1}$$

where the estimate $\tilde{H}(M_{k-1}, X_k)$ and its variance $\hat{\sigma}^2(M_{k-1}, X_k)$ are constructed from the M_{k-1} samples gathered in the earlier iteration. The expression in (4.1) can be interpreted as the result of balancing the squared bias and the variance of the function estimator; it can also be interpreted as the minimum sample size required to declare with some certainty that the function value $h(X_k)$ at the current iterate X_k has been estimated to a sufficient level of accuracy to rule out X_k being the solution. [25] show that under the sampling rule (4.1), the resulting iterates converge to a zero of h and the samples M_k grow geometrically. The proof for this convergence requires a strong condition (Eq 4.20), in part because of the two-stage nature of the procedure, and it is unclear how such a condition can be checked a priori.

A competing fully *sequential* rule proposed by [85] has the following form:

$$M_k = \inf_m : \ \frac{\hat{\sigma}(m, X_k)}{\sqrt{m}} < \alpha \|\tilde{H}(m, X_k)\|, \quad \alpha > 0, \tag{4.2}$$

(A simpler version of (4.2) was proposed in [5] within the context of estimating a confidence interval on the mean.) [85] conjecture that the use of the fully sequential stopping rule (4.2) in (SCSR) results in convergent and asymptotically efficient iterates.

4.1 Contributions

We investigate the use of adaptive sampling within stochastic recursions (SCSR) for solving SO and SRFPs. The adaptive sampling schemes we introduce are a fully sequential version of (4.1), and are constructed to balance the estimated variance and squared bias of the (recursive) function estimates at each visited point. There is emerging evidence that schemes similar to what we propose work well in practice and come closer to the goal of achieving robust finite-time performance with no user-intervention. However, the analysis of such fully adaptive schemes turns out to be challenging, and there appears to be no clear analysis of the consistency and efficiency of the resulting iterates to date. In this chapter, we first present two results on *fixed-width* sequential sampling schedules in Section 4.3 that take us closer to the construction of provably consistent and efficient adaptive sampling schemes within stochastic recursions:

(1) We first analyze a simple sampling rule similar to (4.2) obtained by replacing the $\tilde{H}(m, X_k)$ on the right-hand side with a geometrically decreasing deterministic sequence γ^k, for some $\gamma \in (1, \bar{\gamma})$, where $\bar{\gamma}$ is defined based on some prior curvature information . We show that under such a rule, the iterates converge efficiently. This result is a slight generalization of the geometric sample growth rates that are shown to be efficient in SCSRs (Chapter 3), allowing the sample sizes m_k to also react to local estimation conditions (i.e. $\hat{\sigma}(m, x_k)$).

(2) We next analyze a version of the sampling rule (4.2) that replaces $\tilde{H}(m, X_k)$ on the right-hand side with the actual function value $h(X_k)$. Our proposed scheme adapts to the local conditions of the iterations of SCSR by determining the amount of sampling needed solely based on the relative accuracy of the function estimate at the current iterate. We start with known results for sequential estimation methods, first described by [27] for iid populations with unknown variance, and extend their analysis to show that (SCSR) augmented with the proposed sequential sampling rule is asymptotically efficient.

Adaptive-sampling schemes are then proposed under the context of *relative-width* sequential sampling schedules, by preserving $\tilde{H}(m, X_k)$ on the right-hand side of the sampling rule (4.2). (SCSR) augmented with such sequential sampling schemes is called "Adaptive-SCSR". Given different types of estimator, Adaptive-SCSR is proved to be strongly consistent and efficient in Section 4.4.

4.2 Assumptions

We place the following standing conditions on the function $h(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ of interest to the SRFP problem of determining a x^* that satisfies $h(x^*) = \bar{0}$.

Assumption 4.2.1. *The function $h(x)$ satisfies the following:*

A1 There exists a unique x^* such that $h(x^*) = \bar{0}$,

A2 for all $x \in X$, $(x - x^*)^T h(x) \geq l_0 \|x - x^*\|^2$,

A3 h is locally Lipschitz continuous at x^*, that is, there exists $l_1 > 0$, such that for all x, $\|h(x)\| \leq l_1 \|x - x^*\|$.

Analogously, the function $f(\cdot) : \mathbb{R}^d \to \mathbb{R}$ of interest to the SO problem $\{\min_x f(x)\}$ is assumed to satisfy the following.

Assumption 4.2.2. *(*Strong Convexity*) The function $f(x)$ is twice continuously differentiable and there exist constants $0 < \lambda < \beta$, such that*

$$\lambda \|u\|^2 \leq u^T \nabla h(X_k) u \leq \beta \|u\|^2, \text{ for all } X \text{ and } u. \tag{4.3}$$

In the context of SRFPs, the function h of Assumption 4.2.1 relates to the function f of Assumption 4.2.2 as $\nabla f(x) = h(x)$, and the conditions can be verified to yield the same SCSR error structure.

Eventually, we call a stochastic recursion asymptotically efficient if the following holds.

Definition 4.2.1. (Asymptotic Efficiency) *Denote $\Gamma_k := \sum_{i=1}^{k} M_i$ as the total samples used up till the kth iteration. If there exists a sequence $\{\nu_k\}$ such that $\Gamma_k = O_p(\nu_k)$, then a stochastic recursion with iterations defined by (SCSR) converges asymptotically efficiently if*

$$\mathbb{E}[h(X_k)] \;=\; \mathbb{E}[h(X_k) - h(x^*)] \;=\; O(\nu_k^{-1}). \tag{4.4}$$

In Chapter 3 Theorem 3.5.3 showed that this rate is the fastest that any sampling-controlled stochastic recursion (SCSR) under the assumed conditions can achieve. We anticipate that this result will be true for stochastic recursions with dynamic random sample sizes M_k.

4.3 Main Results: Fixed Width Sequential Sampling Schedule

In this section we investigate the behavior of sampling-controlled stochastic recursions

$$X_{k+1} = X_k - \frac{1}{\beta}\tilde{H}(M_k, X_k) \tag{SCSR}$$

when augmented with sequential rules for choosing the sample size M_k. Under each of the two rules we consider, the sample size is, conditional on the current iterate X_k, a random variable. We shall use the notation M_k to emphasize this distinction. The random variable M_k is a stopping time adapted to the sequence $\{Y_i\}$ in both methods, with the natural filtration

$$\begin{aligned} \mathcal{F}_k \quad &= \quad \{x_0, (M_1, \Xi_{M_1}), (M_2, \Xi_{M_2}), \ldots, (M_{k-1}, \Xi_{M_{k-1}})\}; \\ &\text{where} \quad \Xi_j = (Y_1, Y_2, \ldots, Y_j),\ j = 1, \ldots, M_{k-1}, \end{aligned} \tag{4.5}$$

and is determined as the lowest sample size that matches the confidence interval of the estimate $\tilde{H}(M_k, X_k)$ to a target value. A measure of the squared half-width of the confidence interval is $\hat{\sigma}^2(M_k, X_k)/M_k$.

The main idea underlying our adaptive sampling proposal is to continue sampling at a point until there is enough probabilistic evidence that the subsequent iterate X_{k+1} is of a higher quality (in terms of objective function value) than the current iterate X_k. The corresponding sample size M_k will then be used in estimating the objective function and its derivatives at the incumbent point. The sample size determining rules M_k are designed to provide the stochastic recursion the flexibility to adapt to the problem structure and exhibit both good performance in finite time and asymptotic efficiency.

Our sequential sampling rules are motivated by those studied in the context of sequential statistics, where stopping rules are defined to estimate the true mean of a given population with either known or unknown distribution [28, 102, 103, 72]. Within the fixed-width sequential stopping rules, a prescribed small width (relative to the true variance of the population) is defined for the confidence interval on the unknown mean of the population. Of performance metric considered in such contexts is the coverage probability; that is the probability of covering the true mean given the prescribed width for the confidence interval. When the variance is known, it is shown that central limit theorem suggests the sample size that guarantees the desired coverage. In case of unknown variance, sequential sampling rules suggest asymptotic rate for the sample size to achieve zero "expected loss"

asymptotically. The loss function is usually proportional to the mean squared error with respect to the true mean of the population. The stopping rule usually balance the unbiased estimator of the standard error with the width of the confidence interval. It is known that the sampling stopping rule is well-defined if for any positive value of the width of the confidence interval, the stopping rule yields a finite sample size. However when the width is tending to zero, the stopped sample size needs to diverge to infinity so as to cover the exact true mean and furnish asymptotic consistency. Hence the "limiting behavior" of the stopped process and of its first moment result in asymptotic consistency and "risk efficiency" of the sampling rule.

Remark 4.3.1. *The sequential sampling stopping rule for estimating the mean of the population with unknown variance, is defined to be asymptotically consistent if as the width of the confidence interval tends to zero, (i) the coverage probability tends to one and (ii) the rate of the resulted sample size be the same as of the case where the variance is known; in other words, with a sufficiently small value of the width, the stopped sample size is of the same order as that of deterministic sample size.*

Remark 4.3.2. *The sequential sampling stopping rule for estimating the mean of the population with unknown variance, is defined to be risk efficient if the expected loss calculated according to the stopped process, is behaving equally as the expected loss with known variance. In other words , when the width of the confidence interval tends to zero, the stopping rule is called risk-efficient, if the fraction of the expected loss in the two cases tends to one.*

The analysis of risk-efficiency and consistency is usually much simpler under parametric assumptions on the population, e.g. normality of the observations. Under such assumptions the loss function becomes independent of the stopped process, and this simplifies the expansion for the expected loss when the variance is unknown. For instance [102, 103] demonstrate asymptotic efficiency of sampling stopping rules, under the sole condition that the sampling starts with three observations. In particular the starting sample size needs to be large relative to the moment index of the error that is considered within the loss function. Otherwise the fraction of loss functions for analysis of risk efficiency would diverge. In general, it is known that when the type of the distribution is known, the minimal rate of the sample size can be always replaced with just a fixed positive integer [28]. However [28] bring an example for inefficiency of the sampling rule under general assumptions on the population. They show that the expected loss with stochastic sampling goes to zero slower than the expected loss with deterministic sampling. Accordingly in the absence of parametric assumptions, "stringent delay

factors" [28] or "minimal rate functions" of sample size [72] are required to handle the analysis.

Within the "fixed-width sequential sampling schedules" proposed in this section, under parametric/non-parametric assumptions on the population, the width of the confidence interval is matched to a *pre-specified* sequence of target values. For the first proposed schedule, this sequence is denoted by γ_k, for which we assume $\gamma_k \to 0$ and $k \to \infty$, but $\sum_k \gamma_k < \infty$. Theorem 4.3.1 shows that under these mild conditions on γ_k, the sequence of iterates $\{X_k\} \overset{\text{wp1}}{\to} x^*$. Furthermore, if γ_k were to grow geometrically, then the recursion is asymptotically work-efficient for all geometric growth factors up to a finite upper bound. The main tool used for analysis of the proposed sequential sampling rules embedded in stochastic recursions, is to study asymptotic theories for randomly stopped random sequences, traces back to [5].

For an easy exposition of the stated theorems in this section, we first rigorously define what we mean by the estimate of the function $h(x)$, required within (SCSR).

Assume that i.i.d. observations $Y_1, Y_2, \ldots$ on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$ are available such that their (unknown) mean $\mathbb{E}[Y_i(x)] = h(x)$, $\mathbb{E}[Y_i^2(x)] < \infty$, and the (unknown) non-singular $(d \times d)$-covariance matrix $\Sigma(x)$, with

$$\sigma^2 = \sup_x \operatorname{tr}(\Sigma(x)) < \infty. \tag{4.6}$$

The estimator for $h(x)$ from m copies of these observations is

$$\tilde{H}(m, x) = m^{-1} \sum_{i=1}^{m} Y_i(x).$$

The following linear transformation of the sample covariance matrix will play an important role in the sequel:

$$\hat{\sigma}^2(m, x) = \operatorname{tr}(\frac{1}{m-1} \sum_{i=1}^{m} (Y_i - \tilde{H}(m, x))(Y_i - \tilde{H}(m, x))^T).$$

Theorem 4.3.1. *Let $\{\gamma_k\}_{k \geq 1}$ be a fixed positive sequence for which we have $\sum_{i=1}^{\infty} \gamma_i < \infty$. Let the function $h(\cdot)$ satisfy Assumption 4.2.1.*

(i) Suppose for $\alpha > 0$, β in (SCSR) satisfy $\frac{(1+\alpha)l_1^2}{2l_0} < \beta < \infty$. Denote $\{M_k\}_{k \geq 1}$ as a sequence of random variables in the probability space $(\Omega, \mathcal{F}, \mathbb{P})$.

Assume either of the following holds.

(R1) Y_1 *is normally distributed, and* $M(X_k) = \inf\{m > 3 : \frac{\hat{\sigma}^2(m,X_k)}{m} < \alpha\gamma_k|\mathcal{F}_k\}$, $k = 1, 2, \ldots$;

(R2) $\mathbb{E}[Y_1^6] < \infty$, *and for* $0 < \alpha < 1$, $M(X_k)|\mathcal{F}_k = \inf\{m > max(2, [\alpha\gamma_k]^{-1/2} + 1) : \frac{\hat{\sigma}^2(m,X)}{m} < \alpha\gamma_k|\mathcal{F}_k\}$.

Then (SCSR) satisfies $X_k \overset{wp1}{\to} x^*$.

(ii) Further, letting $\gamma_k^{-1} = [\gamma]^k$, $1 < \gamma \leq (1 - \frac{2l_0}{\beta} + \frac{(1+\alpha)l_1^2}{\beta^2})^{-1}$, *the algorithm is asymptotically efficient.*

The following lemma is used in proving Theorem 4.3.1.

Lemma 4.3.1. *Let* $Y_i s$, $i = 1, \ldots$ *be iid observations from* $N(\mu, \Sigma)$, *whose m-sample mean is denoted by* $\mathcal{Z}_m = \frac{1}{m}\sum_{i=1}^m Y_i$. *Consider the following sequential procedure:*

$$M_c = \inf\{m \geq 1 : \frac{\hat{\sigma}^2(m)}{m} < c\}, \tag{4.7}$$

where σ_m^2 *is the trace of the sample covariance matrix, c is a positive constant that is allowed to approach zero. Letting* $\sigma^2 = tr(\Sigma)$, *we have*

(i) M_c *is a stopping time with respect to* $\{Y_i\}_{1\leq i\leq m}$, *and when* $c \to 0$, $(\frac{c}{\sigma})^2 M_c \overset{wp1}{\to} 1$;

(ii) for the stopped process $\mathcal{Z}_{M_c}$ *we have* $Var(\mathcal{Z}_{M_c}) = \mathbb{E}[\sigma^2 M_c^{-1}]$.

Proof. First we prove that M_c is a well-defined stopping time with respect to $\{Y_i\}_{1\leq i\leq m}$. Consider the stochastic process $Y = \{Y_m : m \in \mathbb{N}\}$ defined on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$. We note that M_c as defined in (4.7) is a random time for the stochastic process $Y = \{Y_m : m \geq 1\}$, as M_c is a discrete random variable on the same probability space as Y. For $m \in \mathbb{N}$, let $\mathcal{F}_m = \sigma\{Y_s, s \in \mathbb{N}, s \leq m\}$, be the σ-algebra of events up to time m. The random time M_c, is a stopping time since

$$\{M_c > m\} \in \mathcal{F}_m,$$

for each $m \in \mathbb{N}$. In other words, $\{M_c = m\}$, is $\mathcal{F}_m$ measurable, which means that the event $\{M_c = m\}$ is completely determined by the total information up to time m, $\{Y_1, Y_2, \ldots, Y_m\}$, and is not dependent on the future $Y_{m+1}, Y_{m+2}, \ldots$.

Then part (i) follows by Lemma 2 in [27]. For the second part, we note that the probability distribution of M_c is defined for any $m \geq 1$ by

$$P(M_c = m) = P\{\hat{\sigma}_k^2 < ck \text{ for } k = m \text{ but not for any } k < m\}. \tag{4.8}$$

Since Y_is are normally distributed, we know from [10] that $\hat{\sigma}^2(m)$ is statistically independent of $\mathcal{Z}_m$. Therefore

$$P(M = m|\mathcal{Z}_M) = P(M = m). \tag{4.9}$$

Hence the event $\{M = m\}$ is independent of $\mathcal{Z}_M$, and so

$$\begin{aligned} \mathrm{Var}(\mathcal{Z}_M) &= \mathbb{E}[\mathrm{Var}(\mathcal{Z}_M|M = m)] + \mathrm{Var}(\mathbb{E}[\mathcal{Z}_M|M = m]), \\ &= \mathbb{E}[\sigma^2 M^{-1}]. \end{aligned}$$

□

Proof of Theorem 4.3.1(i). First we find a finite time upper bound on the squared error. To this end, by (SCSR), letting $Z_k = X_k - x^*$, we have for all k,

$$\begin{aligned} Z_{k+1}^2 &= Z_k^2 - \frac{2}{\beta} Z_k^T \tilde{H}(M_k, X_k) + \frac{1}{\beta^2}\|\tilde{H}(M_k, X_k)\|^2, \\ &= Z_k^2 - \frac{2}{\beta} Z_k^T (\tilde{H}(M_k, X_k) - h(X_k)) - \frac{2}{\beta} Z_k^T h(X_k) + \frac{1}{\beta^2}\|\tilde{H}(M_k, X_k) - h(X_k)\|^2 \\ &\quad + \frac{1}{\beta^2}\|h(X_k)\|^2 + \frac{2}{\beta^2} h(X_k)^T (\tilde{H}(M_k, X_k) - h(X_k)). \end{aligned}$$

By Assumption 4.2.1(A2), we have

$$\begin{aligned} \mathbb{E}_\Omega[Z_{k+1}^2|\mathcal{F}_k] &= Z_k^2 - \frac{2}{\beta} Z_k^T h(X_k) + \frac{1}{\beta^2}\|h(X_k)\|^2 + \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k], \\ &\leq Z_k^2 - \frac{2l_0}{\beta} Z_k^2 + \frac{l_1^2}{\beta^2} Z_k^2 + \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k], \\ &= (1 - \frac{2l_0}{\beta} + \frac{l_1^2}{\beta^2}) Z_k^2 + \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k]. \end{aligned} \tag{4.10}$$

Letting $a := 1 - \frac{2l_0}{\beta} + \frac{l_1^2}{\beta^2}$, since $\beta > \frac{l_1^2}{2l_0}$, $0 < a < 1$.

Under (R1), by Lemma 4.3.1 part (ii) we have

$$\mathbb{E}_\Omega[(\tilde{H}(M_k, X_k) - h(X_k))^2|\mathcal{F}_k] = \mathrm{tr}(\Sigma)\mathbb{E}_\Omega[M_k^{-1}|\mathcal{F}_k].$$

Hence by (4.21) and Theorem 3_* of [102], when $\gamma_k \to 0$,

$$\begin{aligned} \mathbb{E}_\Omega[Z_{k+1}^2|\mathcal{F}_k] &\leq a Z_k^2 + \frac{\sigma^2}{\beta^2}\mathbb{E}_\Omega[M_k^{-1}|\mathcal{F}_k], \\ &\leq a Z_k^2 + \frac{\alpha}{\beta^2}\gamma_k. \end{aligned} \tag{4.11}$$

Since $\sum_{i=1}^{\infty} \gamma_k < \infty$, by Lemma 2 in [118], we have that $X_k \overset{\text{wp1}}{\to} x^*$.

Under (R2), by (4.21) and the relation (2.4) in [72], (4.11) holds, and the claim follows accordingly.

Proof of Theorem 4.3.1(ii)

By Assumptions 4.2.1(A2) and (A3), we have

$$l_0^2 z_k^2 \leq \|h(X_k)\|^2 \leq l_1^2 z_k^2. \tag{4.12}$$

Considering $\Gamma_k = O_p(\sum_{i=1}^{k}[a + \alpha(\frac{l_1}{\beta})^2]^{-i})$, asymptotic efficiency of (SCSR) follows by (4.11), corollary 4.3 of [25] and Definition 4.2.1. □

Theorem 4.3.1 closely matches a result for SCSRs under the same conditions; in Chapter 3 we showed that pre-determined sample sizes m_k that grow geometrically with the same growth rate restrictions as Theorem 4.3.1(ii) are asymptotically efficient. Note that the sequential sampling rule introduced in Theorem 4.3.1 results in larger samples than the lower bound of the geometrically growing γ_k^{-1}, and is sensitive to the quality of the estimator $\tilde{H}(m, x)$ at the current estimate. This sequential stopping rule is easy to implement given the chosen sequence $\{\gamma_k^{-1}\}$ since the update of the variance estimator $\hat{\sigma}^2$ is a constant-computational-effort operation.

We note that the random sample size $M_k \sim O_p(\gamma_k^{-1})$, and the sequences $\{\gamma_k^{-1}\}$ that were judged efficient, grow exactly as the inverse of $E[h(X_k)]$ by Theorem 4.3.1(ii). Although we do not yet have a truly hands-off method since the user needs to still pick the geometric growth factor γ carefully to ensure efficiency, the results are motivating the next sampling rule 4.13, where the idea is to sample till the sampling error at the current iterate is just smaller than the optimality gap of the iterate:

For $\alpha > 0$,

$$M_k = \inf\{m > \max\{3, \gamma_k\} : \frac{\hat{\sigma}^2(m, X_k)}{m} < \alpha\|h(X_k)\|^2 | \mathcal{F}_k\}. \tag{4.13}$$

This fixed width sequential sampling rule is replacing the sequence $\{\gamma_k\}$ with purely local information. Theorem 4.3.2 analyses the asymptotic behavior of the corresponding SCSR method.

Theorem 4.3.2. *The function $h(\cdot)$ satisfies Assumption 4.2.1. Let $\{\gamma_k\}_{k\geq 1}$ be a fixed positive sequence for which we have $\sum_{i=1}^{\infty} 1/\gamma_i < \infty$. Denote $\{M_k\}_{k\geq 1}$ as a sequence of random variables in probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Given $\mathcal{F}_k$ defined in (4.5), assume either of the following conditions hold.*

(C1) (Parametric Setup) Y_1 *is normally distributed, and for* $0 < \alpha < 1$, $M_k := M(X_k|\mathcal{F}_k) = \inf\{m > max\{3, \gamma_k\} : \frac{\hat{\sigma}^2(m,X_k)}{m} < \alpha\|h(X_k)\|^2|\mathcal{F}_k\}$;

(C2) (Nonparametric Setup) Let $\mathbb{E}[Y_1^8] < \infty$, *and for* $0 < \alpha < 1/4$, $\zeta > 0$, $M_k := M(X_k) = \inf\{m > max\{3, \gamma_k\} : \frac{\hat{\sigma}(m,X_k)}{m} + m^{-(1+\alpha)} < \zeta\|h(X_k)\|^2|\mathcal{F}_k\}$.

Then the SCSR iterates (SCSR) are (a) almost surely convergent to the true solution x^*, *and (b) asymptotically efficient.*

Proof. First we note that under either $(C1)$, or $(C2)$, by Lemma 4.3.1, M_k is a stopping time with respect to $\mathcal{F}_k$.

Under $(C1)$, with probability one convergence follows by slight changes in Theorem 4.3.1. For efficiency of (SCSR), by (4.11) we get

$$\mathbb{E}_\Omega[Z_{k+1}^2|\mathcal{F}_k] \;\leq\; aZ_k^2, \tag{4.14}$$

where $a := 1 - \frac{2l_0}{\beta} + \frac{(1+\alpha)l_1^2}{\beta^2}$, and by $\beta > \frac{(1+\alpha)l_1^2}{2l_0}$, $0 < a < 1$. Hence letting $b_k := \mathbb{E}[Z_k^2]$, $q_k := \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2]$ and $d_k := b_1(1 - \frac{2l_0}{\beta})^k + \sum_{i=2}^{k-1}(1 - \frac{2l_0}{\beta})^{k-i}q_i + q_k$, and

$$\eta = \max(b_{k_0}a^{1-k_0}, \max_{1\leq k\leq k_0}\{a^{-k}d_k\}),$$

for all $k \geq 1$, we have $b_k \leq \eta a^k$.

Since for all k, $b_k \leq a\eta$, Z_k^2 is uniformly integrable, and so is $\|h(X_k)\|^2$, by (4.36). Hence $\mathbb{E}\|h(X_k)\|^2$ approaches to zero with the same geometric rate as b_k.

Moreover since for all k, $\Pr(M_k < \infty) = 1$, we have

$$\begin{aligned}\mathbb{E}[\|h(X_k)\|^2 M_k] &= \mathbb{E}[\mathbb{E}[\|h(X_k)\|^2 M_k|\mathcal{F}_k]],\\ &= \mathbb{E}\|h(X_k)\|^2\mathbb{E}[M_k|\mathcal{F}_k].\end{aligned} \tag{4.15}$$

Thus, when for a given ε, $\|Z_k\| \leq \varepsilon$, $\mathbb{E}\|h(X_k)\|^2 M_k \overset{\text{wp1}}{\to} \sigma^2/\alpha$ and we have

$$\frac{\log M_k}{k} = \frac{\log M_k\mathbb{E}\|h(X_k)\|^2}{k} - \frac{\log\mathbb{E}\|h(X_k)\|^2}{k} \approx \frac{\log 1/b_k}{k} \to 1/a,$$

which proves that as $X_k \to X^*$, M_k is geometrically growing with constant $1/a$.

Accordingly, asymptotic efficiency of the method follows by considering

$$\Gamma_k = O_p(\sum_{i=1}^{k}[a]^{-i}) = O_p([a]^{-k}),$$

and $\nu_k = [a]^k$ in Definition 4.2.1.

Under $(C2)$, first we prove that (SCSR) is convergent with probability one. Letting $c_k := \alpha\|f(X_k)\|^2$, for all iterations we have

$$P(M_k = \infty) = \lim_{m\to\infty} P\{M_k > m\} \leq P\{\sigma_m^2(X_k)/m > c_k \text{ for all } m \geq \gamma_k\} = 0 \quad (4.16)$$

as σ_m^2 is convergent with probability one as $m \to \infty$. Therefore $P\{M_k < \infty\} = 1$, and

$$\sqrt{M_k}(\|\tilde{H}(M_k, X_k) - h(X_k)\|) \leq \sup_m \sqrt{m}(\|\tilde{h}_m(X_k) - h(X_k)\|).$$

Besides since $\mathbb{E}[\|Y_1\|^2] < \infty$, $\mathbb{E}[\sup_m \ m\|\tilde{h}_m(X_k) - h(X_k)\|^2] < \infty$; together with

$$\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k] \leq \frac{1}{\gamma_k}\mathbb{E}[(\sqrt{M_k}(\|\tilde{H}(M_k, X_k) - h(X_k)\|))^2|\mathcal{F}_k],$$

by 4.21, $\sum_k \gamma_k^{-1} < \infty$ and Lemma 2 in [118], we conclude SCSR is convergent with probability one.

In order to prove part (ii) of the theorem for condition $(C2)$, we first note that by [45], when $\theta_k \to x^*$, $\|\tilde{H}(M_k, X_k) - h(X_k)\|^2\|h(X_k)\|^{-2}$ is uniformly integrable and we have

$$\mathbb{E}\|\tilde{H}(M_k, X_k) - h(X_k)\|/\|h(X_k)\| \to \zeta.$$

Therefore by (A3) and (4.21), there exists k_0, such that for all $k > k_0$,

$$\begin{aligned}
\mathbb{E}_\Omega[Z_{k+1}^2|\mathcal{F}_k] &\leq (1 - \frac{2l_0}{\beta} + \frac{l_1^2}{\beta^2})Z_k^2 + \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k], \\
&= (1 - \frac{2l_0}{\beta} + \frac{l_1^2}{\beta^2})Z_k^2 \\
&\quad + \frac{1}{\beta^2}\|h(X_k)\|^2\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2/\|h(X_k)\|^2||\mathcal{F}_k], \\
&\leq (1 - \frac{2l_0}{\beta} + \frac{l_1^2}{\beta^2})Z_k^2 + \frac{\zeta l_1^2}{\beta^2}Z_k^2.
\end{aligned}$$

Accordingly efficiency of (SCSR) follows by the same approach as in part $(C1)$. □

The following corollary is an immediate consequence of Theorem 4.3.2 and the equivalence of Assumption 4.2.1 for $h(\cdot)$ and Assumption 4.2.2 for $f(\cdot)$ where $h(x) = \nabla_x f(x)$.

Corollary 4.3.1. *The function $h(\cdot)$ satisfies the conditions in Assumption 4.2.2. Let $\{\gamma_k\}_{k\geq 1}$ be a fixed positive sequence for which we have $\sum_{i=1}^{\infty} 1/\gamma_i < \infty$. Define $\{M_k\}_{k\geq 1}$ as a sequence of random variables in probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Let $\mathbb{E}[Y_1^8] < \infty$, and for $0 < \alpha < 1/4$, $\zeta > 0$, $M_k := M(X_k) = \inf\{m > max\{3, \gamma_k\} : \frac{\hat{\sigma}(m,X_k)}{m} + m^{-(1+\alpha)} < \zeta \|h(X_k)\|^2 | \mathcal{F}_k\}$. Then (SCSR) is (a) almost surely convergent to the true solution x^*, and (b) asymptotically efficient.*

The version of the sequential rule used in Theorem 4.3.2 and Corollary 4.3.1 has the true function $h(X_k)$ value on the right-hand side of 4.13, thereby leaving no critical parameters for choice by the user. Thus, this version of the stochastic recursion is truly parameter-free and fully adaptive, in that the sample size needed in each iteration is determined solely by local functional and estimation properties. Such a rule is, however, not implementable because the function $h(X_k)$ is not known. To reach a completely adaptive version that can be implemented easily, the results in the next section modify 4.13 to:

$$M_k = \inf\{m > \gamma_k : \frac{\hat{\sigma}^2(m, X_k)}{m^{1-\varepsilon}} < \alpha \|\tilde{H}(m, X_k)\|^2 | \mathcal{F}_k\}, \tag{4.17}$$

where $0 < \varepsilon < 1$, is called a *coercion factor*, whose role is discussed in detail in the next section. This rule is easy to implement as a sequential rule if the estimator $\tilde{H}(m, X_k)$ and the variance function $\frac{\hat{\sigma}^2(m,X_k)}{m^{1-\varepsilon}}$ can be updated using constant-effort computations. The results in Theorem 4.3.2 indicate that one can expect similar convergence properties. Unlike the rule 4.13 which compares the *absolute* confidence interval to a fixed target $h(X_k)$, the rule 4.17 compares the *relative* confidence interval $\hat{\sigma}^2(m, x)/(m^{1-\varepsilon}\|\tilde{H}(m, x)\|^2)$ to a target. Thus, the convergence under 4.17 is not a straightforward consequence of the proof method of Theorem 4.3.2. The convergence properties under the rule 4.17 is discussed in the next section.

4.4 Main Results: Relative Width Sequential Sampling Schedule

In this section too, we investigate asymptotic behavior of (SCSR), but augmented with *relative-width* sequential rules for choosing the sample size M_k, as opposed to the *fixed-width* sequential sampling stopping rules of the previous section. Unlike previous section, the width of the confidence interval is matched to a sequence of target values that is itself sample-path dependent, and not "fixed" any more. Considering $\mathcal{F}_k$ as defined in (4.5), the proposed stopping rule to determine the sample size is as follows:

For $c > 0$, and $0 < \varepsilon < 1/2$, set

$$M_k := \inf\{m > \nu_k \ : \hat{\sigma}(m, X_k)/m^{1/2(1-\varepsilon)} \leq \rho\|\tilde{H}(m, X_k)\||\mathcal{F}_k\}. \quad (4.18)$$

(SCSR) augmented with the above sequential sampling rule is called "Adaptive-SCSR". Under general types of biased/unbiased estimators, asymptotic behavior of Adaptive-SCSR is studied in this section.

4.4.1 Adaptive-SCSR with Unbiased Estimator

The following lemma helps to prove Theorem 4.4.1 on consistency of (SCSR) augmented with (4.19).

Lemma 4.4.1. *Let $\{Z_i, i \geq 1\}$ be a sequence of iid random variables defined on the probability space $(\Omega, \mathbb{F}, \mathbb{P})$ with $\mathbb{E}[Z_1] = 0$ and $\mathbb{E}[Z_1^2] < \infty$. Also let M be a random variable on the same probability space Ω, that is finite almost surely, i.e. for $\omega \in \Omega$, the probability of the event $\{\omega : M(\omega) < \infty\}$ is one; $Pr(M < \infty) = 1$. Setting $\bar{Z}_k := \sum_{i=1}^{k} Z_i/k$, we have $\mathbb{E}[\|\sqrt{M}\bar{Z}_M\|^2] < \infty$.*

Theorem 4.4.1. *Assuming $\sum_k \nu_k^{-1} < \infty$, $c > 0$, $0 < \varepsilon < 1/2$, $\beta > \frac{l_1^2}{2l_0}(1+\xi)$, and $\xi > 0$, set*

$$M_k := \inf\{m > \nu_k \ : \hat{\sigma}(m, X_k)/m^{1/2(1-\varepsilon)} \leq \rho\|\tilde{H}(m, X_k)\||\mathcal{F}_k\}, \quad (4.19)$$

Given $\mathbb{F}_k$, for $X_k \neq x^*$, $P\{M_k < \infty|\mathcal{F}_k\} = 1$, and (SCSR) are almost surely convergent to the true solution x^*.

Proof. For all iterations ($\theta_k \neq 0$) we have

$$\begin{aligned} P\{M_k = \infty|\mathcal{F}_k\| &= \lim_{m\to\infty} P\{M_k > m|\mathcal{F}_k\}, \\ &\leq P\{\hat{\sigma}(m, X_k)/m^{1/2(1-\varepsilon)} > c\tilde{H}(m, X_k)\forall m \geq \nu_k|\mathcal{F}_k\} = 0; \end{aligned}$$

providing that given $\mathcal{F}_k$, $\lim_{m\to\infty} \hat{\sigma}(m, X_k) = \sigma^2$ and $\lim_{m\to\infty} \tilde{H}(m, X_k) = h(X_k)$. Hence M_k is well-defined. We now proceed to prove almost sure convergence of the method. To this end letting $Z_k = X_k - x^*$, by (SCSR) we have for all k,

$$\begin{aligned} Z_{k+1}^2 &= Z_k^2 - \frac{2}{\beta}Z_k^T\tilde{H}(M_k, X_k) + \frac{1}{\beta^2}\|\tilde{H}(M_k, X_k)\|^2, \\ &= Z_k^2 - \frac{2}{\beta}Z_k^T(\tilde{H}(M_k, X_k) - h(X_k)) - \frac{2}{\beta}Z_k^T h(X_k) + \frac{1}{\beta^2}\|\tilde{H}(M_k, X_k) - h(X_k)\|^2 \\ &\quad + \frac{1}{\beta^2}\|h(X_k)\|^2 + \frac{2}{\beta^2}h(X_k)^T(\tilde{H}(M_k, X_k) - h(X_k)). \end{aligned}$$

By [A2], we have

$$\begin{aligned}\mathbb{E}_\Omega[Z_{k+1}^2|\mathcal{F}_k] &= Z_k^2 - \frac{2}{\beta}Z_k^T h(X_k) + \frac{1}{\beta^2}\|h(X_k)\|^2 \\ &\quad + \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k], \qquad (4.20)\\ &\leq Z_k^2 - \frac{2l_0}{\beta}Z_k^2 + \frac{l_1^2}{\beta^2}Z_k^2 + \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k], \\ &= (1 - \frac{2l_0}{\beta} + \frac{l_1^2}{\beta^2})Z_k^2 + \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k]. \qquad (4.21)\end{aligned}$$

Accordingly by $\mathbb{E}[Y_1^2(\theta_k)] < \infty$, $P\{M_k < \infty|\mathcal{F}_k\} = 1$, Lemma 4.4.1, and (4.6)

$$\begin{aligned}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k] &= \mathbb{E}_\Omega[M_k^{-1}M_k\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k], \\ &\leq \frac{1}{\nu_k}\mathbb{E}_\Omega[\|\sqrt{M_k}\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k], \\ &= O(\frac{1}{\nu_k}). \qquad (4.22)\end{aligned}$$

Hence by 4.21, Lemma 2 in [118], and $\sum_k \nu_k^{-1} < \infty$, the stochastic recursion (SCSR) is convergent to x^* with probability one. □

Throughout the following theorems, almost sure convergence of (SCSR) is assumed. Let $\theta_k := \mathbb{E}[h(X_k)|\mathcal{F}_k]$ be the true gradient evaluated at the kth iteration, hence static. By consistency, given $\varepsilon_1 > 0$, there exists $k_1 < \infty$ such that for all $k > k_1$, $\|\theta_k\| < \varepsilon_1$. Also by $\sum_k \nu_k^{-1} < \infty$, there exists k_2 such that for all $k > k_2$, $\nu_k > B$ for B arbitrarily large. Hence given $\mathcal{F}_k$, letting $K := \max\{k_1, k_2\}$, for a $k_0 > K$, θ_{k_0} is arbitrarily close to zero , M_{k_0} is arbitrarily large, and given ε_2, arbitrarily small,

$$\|\hat{\sigma}^2(m, X_{k_0}) - \mathrm{tr}(\Sigma(X_{k_0}))\| < \varepsilon_2, \text{ for } m \geq \nu_{k_0}. \qquad (4.23)$$

Setting $k := k_0 > K$, we outline Theorem 4.4.2 and 4.4.3. Before setting up the main results, we outline two main lemmas.

Lemma 4.4.2. *Let $\bar{Z}(m, X_k)$ be the unbiased sample average of m iid observations at X_k with mean zero and variance σ^2. Denote*

$$\begin{aligned}T_k &:= \{\nu_k, \nu_k + 1, \ldots, \theta_k^{-\eta}\}; \\ \bar{m}_k^* &:= \arg\max_{m \in T(\theta_k)}\{\frac{m^{1-2\varepsilon}\bar{Z}^2(m, X_k)}{\sigma^2} + m^{1-2\varepsilon}\theta_k^2\}; \\ v_k^* &:= \sup_{m \in T(\theta_k)}\{\frac{m^{1-2\varepsilon}\bar{Z}^2(m, X_k)}{\sigma^2} + m^{1-2\varepsilon}\theta_k^2\}.\end{aligned}$$

Then we have as $k \to \infty$,

(i) $\frac{\tilde{m}_k^*}{\theta_k^{-\eta}} \overset{wp1}{\to} 1.$

(ii) $v_k^* \overset{wp1}{\to} 1.$

Proof. Since $\{\nu_k\} \to \infty$, we notice that

$$\frac{m_k^{1-2\varepsilon}\bar{Z}^2(m_k, X_k)}{\sigma^2} \overset{\text{wp1}}{\to} 0 \text{ as } k \to \infty \tag{4.24}$$

for any sequence $\{m_k\}$ satisfying $m_k \in T_k$. Let the set when (4.24) holds be denoted by $\mathcal{D}$, and so we see that $\Pr(\mathcal{D}) = 1$.

Now suppose (i) is false, i.e. , there exists a set $\mathcal{C}$ of positive measures such that for any sequence $\{\tilde{m}_k(w)\}$, $w \in \mathcal{C}$, there exists a subsequence $\{\tilde{m}_{k_j}(w)\}$, $w \in \mathcal{C}$, satisfying $\frac{\tilde{m}_{k_j}(w)}{\theta_{k_j}^{-\eta}} < 1 - \delta$ for large enough j and some $\delta > 0$.

For such a sequence $\tilde{m}_{k_j}(w)$, we see that for $w \in \mathcal{D} \cap \mathcal{C}$,

$$\frac{(\tilde{m}_{k_j}(w))^{1-2\varepsilon}\bar{Z}^2(\tilde{m}_{k_j}(w), X_k)}{\sigma^2} + \tilde{m}_{k_j}(w)^{1-2\varepsilon}\theta_k^2 < 1 - \delta, \tag{4.25}$$

as $j \to \infty$. (4.25) is a contradiction however because $m_k = \theta_k^{-\eta}$ satisfies

$$\frac{m_k^{1-2\varepsilon}\bar{Z}^2(m_k, X_k)}{\sigma^2} + m_k^{1-2\varepsilon}\theta_k^2 = 1,$$

for $w \in \mathcal{C}$, and as $k \to \infty$. This concludes that (i) holds. Also (ii) holds if (i) holds. □

Lemma 4.4.3. *Suppose for a positive sequence of random variables $\{V_k\}_{k\geq 1}$, $0 < a_1 < a_2 < 1$, we know that $a_1^k \leq \mathbb{E}[V_k] \leq a_2^k$, for all k. Also suppose that V_k is $\mathcal{F}_k$ measurable, i.e. $\mathbb{E}[V_k|\mathcal{F}_k] = V_k$. Now let $\{M_k\}_{k\geq 1}$ be a sequence of random variables, for which we have*

(i) *$Pr(M_k < \infty) = 1$, for all k;*

(ii) $\lim_{k\to\infty} M_k = \infty$;

(iii) *letting $M_k^* := M_k|\mathcal{F}_k$,* $\lim_{k\to\infty} \sup \frac{M_k^*}{\mathbb{E}[M_k^*]} < \infty$.

Also suppose for ρ_1 and ρ_2 positive we have

(A1) $\lim_{k\to\infty} \sup \mathbb{E}[V_k M_k|\mathcal{F}_k] < c_1$, *for* $c_1 > \rho_1$;

(A2) $\lim_{k\to\infty} \inf \mathbb{E}[V_k M_k|\mathcal{F}_k] > c_2$, *for* $0 < c_2 < \rho_2$.

Then

$$\Gamma_k^* \mathbb{E}[V_k] = O(1),$$

where $\Gamma_k = \sum_{i=1}^k M_i^*$.

Proof. Since $\mathbb{E}[V_k] \leq a_2^k$, V_k is uniformly integrable. Therefore by (A1)

$$\lim_{k\to\infty} \sup a_1^k \mathbb{E}[M_k|\mathcal{F}_k] \leq \lim_{k\to\infty} \sup \mathbb{E}[V_k]\mathbb{E}[M_k|\mathcal{F}_k] < c_1.$$

(Therefore $\mathbb{E}[M_k|\mathcal{F}_k]$ should not be faster than geometric.)

Similarly,

$$\lim_{k\to\infty} \inf a_2^k \mathbb{E}[M_k|\mathcal{F}_k] \geq \lim_{k\to\infty} \inf \mathbb{E}[V_k]\mathbb{E}[M_k|\mathcal{F}_k] > c_2.$$

(Therefore $\mathbb{E}[M_k|\mathcal{F}_k]$ should not be slower than geometric.)

Hence $\mathbb{E}[M_k|\mathcal{F}_k] = \Psi(a_2^{-k})$. Then we get

$$\mathbb{E}[\Gamma_k^*] = \sum_{i=1}^k \mathbb{E}[M_i|\mathcal{F}_i] = \Psi(\mathbb{E}[M_k^*]) = \Psi(a_2^{-k}),$$

and so

$$\mathbb{E}[\Gamma_k^*]\mathbb{E}[V_k] = O(1).$$

But assuming $\lim_{k\to\infty} \sup \frac{M_k^*}{\mathbb{E}[M_k^*]} < \infty$:

$$\Gamma_k^* = \sum_{i=1}^k M_i^* = \Psi(\sum_{i=1}^k \mathbb{E}[M_i^*]) = \Psi(\mathbb{E}[\Gamma_k^*]).$$

Therefore

$$\Gamma_k^* \mathbb{E}[V_k] = O(1).$$

□

The next two theorems characterize distribution of the sample size M_k, and show that it "concentrates" around $\theta_k^{-\eta}$ in probability and expectation.

Theorem 4.4.2. *Denote* $\eta := 2/1 - 2\varepsilon$.

(i) $Pr\{\theta_k^\eta M_k \leq x|\mathcal{F}_k\} \leq \exp(-\theta_k^{-4\eta\varepsilon}(1/16\rho^2 - \frac{x^{1-2\varepsilon}}{4\sigma^2})^2)$, $x < (\sigma/2\rho)^\eta$;

(ii) $Pr\{\theta_k^{\eta} M_k > x|\mathcal{F}_k\} \leq \exp(-\theta_k^{-2\eta\varepsilon}\frac{(-x^{1/2-\varepsilon}/\sigma+2/\rho)^2}{2x^{2\varepsilon}}),\ x > (2\sigma/\rho)^{\eta}.$

Proof. We have

$$\Pr\{M_k \leq \theta_k^{-\eta} x|\mathcal{F}_k\} = \Pr\{\sup_{m\in T(\theta_k)} m^{1-2\varepsilon}\frac{\|\tilde{H}(m,X_k)\|^2}{\hat{\sigma}^2(m,X_k)} \geq 1/\rho^2|\mathcal{F}_k\} \quad (4.26)$$

where $T(\theta_k) := \{\nu_k, \nu_k+1, \ldots, \theta_k^{-\eta}x\}$. Now given $\mathcal{F}_k$, $\tilde{H}^2(m,X_k) = \theta^2 + \bar{Z}^2(m,X_k) + 2\theta_k\bar{Z}(m,X_k) \leq 2\theta^2 + 2\bar{Z}^2(m,X_k)$. Let $\bar{m}_k^* := \arg\max_{m\in T(\theta_k)}\{\frac{m^{1-2\varepsilon}\bar{Z}^2(m,X_k)}{\sigma^2} + m^{1-2\varepsilon}\theta_k^2/\sigma^2\}$. By (4.23) we have

$$\begin{aligned}
\Pr\{M_k \leq \theta_k^{-\eta}x|\mathcal{F}_k\} &\leq \Pr\{\sup_{m\in T(\theta_k)}\frac{m\bar{Z}^2(m,X_k)+m\theta_k^2}{\hat{\sigma}^2(m,X_k)m^{2\varepsilon}} \geq 1/2\rho^2|\mathcal{F}_k\}, \quad (4.27)\\
&= \Pr\{\frac{\bar{m}_k^*\bar{Z}^2(\bar{m}_k^*,X_k)}{\sigma^2(\bar{m}_k^*)^{2\varepsilon}} + (\bar{m}_k^*)^{1-2\varepsilon}\theta_k^2/\sigma^2) \geq 1/4\rho^2|\mathcal{F}_k\},\\
&\leq \Pr\{\frac{\bar{m}_k^*\bar{Z}^2(\bar{m}_k^*,X_k)}{\sigma^2} \geq \theta_k^{-2\eta\varepsilon}(1/4\rho^2 - \frac{x^{1-2\varepsilon}}{\sigma^2})|\mathcal{F}_k\},
\end{aligned}$$

where the last step follows by Lemma 4.4.2. By Donsker-Prokhorov invariance principale,

$$\sqrt{m}\bar{Z}(m,X_k)/\sigma \Rightarrow W,$$

where W denotes a standard Brownian Motion(BM). Noting that $1/4\rho^2 > \frac{x^{1-2\varepsilon}}{\sigma^2}$, by Doob's martingale inequality for BM we have

$$\begin{aligned}
\Pr\{M_k \leq \theta_k^{-\eta}x|\mathcal{F}_k\} &\leq \Pr\{W \geq \theta_k^{-2\eta\varepsilon}(1/4\rho^2 - \frac{x^{1-2\varepsilon}}{\sigma^2})|\mathcal{F}_k\},\\
&\leq \exp(-\theta_k^{-4\eta\varepsilon}(1/16\rho^2 - \frac{x^{1-2\varepsilon}}{4\sigma^2})^2|\mathcal{F}_k\},
\end{aligned}$$

which proves part (i). In order to prove part (ii), we calculate $\Pr\{\theta_k^{\eta}M_k > x|\mathcal{F}_k\}$, for $x > 0$.

$$\begin{aligned}
\Pr\{\theta_k^{\eta}M_k > x|\mathcal{F}_k\} &= \Pr\{M_k > x\theta_k^{-\eta}|\mathcal{F}_k\},\\
&= \Pr\{\sup_{m\in T(\theta_k)} m^{1-2\varepsilon}\frac{\|\tilde{H}(m,X_k)\|^2}{\hat{\sigma}^2(m,X_k)} \leq 1/\rho^2|\mathcal{F}_k\},\\
&= \Pr\{\sup_{m\in T(\theta_k)}\frac{\|\sqrt{m}\bar{Z}(m,X_k)+\sqrt{m}\theta_k\|}{m^{\varepsilon}\hat{\sigma}(m,X_k)} \leq 1/\rho|\mathcal{F}_k\},\\
&\leq \Pr\{\sup_{m\in T(\theta_k)}\frac{\|\sqrt{m}\bar{Z}(m,X_k)+\sqrt{m}\theta_k\|}{m^{\varepsilon}\sigma} \leq 2/\rho|\mathcal{F}_k\},
\end{aligned}$$

Similar to part (i), by Donsker-Prokhorov invariance principale,

$$\sqrt{m}\bar{Z}(m, X_k)/\sigma \Rightarrow W,$$

where W denotes a standard Brownian motion. On the other hand, when $m \in T(\theta_k)$, $\sup(m^\varepsilon) = x^\varepsilon \theta_k^{-\eta\varepsilon}$. Therefore

$$\begin{aligned}\sqrt{m}\bar{Z}(m, X_k)/\sigma &= m^\varepsilon \frac{m^{1/2-\varepsilon}}{\sigma}\bar{Z}(m, X_k),\\ &\leq x^\varepsilon \theta^{-\eta\varepsilon}\frac{m^{1/2-\varepsilon}}{\sigma}\bar{Z}(m, X_k).\end{aligned}$$

Hence $\sup_m(\sqrt{m}\bar{Z}(m, X_k)/\sigma) \leq x^\varepsilon\theta^{-\eta\varepsilon}\sup_m(\frac{m^{1/2-\varepsilon}}{\sigma}\bar{Z}(m, X_k))$, and by Doob's martingale inequality,

$$\begin{aligned}\Pr\{\theta_k^\eta M_k > x|\mathcal{F}_k\} &\leq \Pr\{\|W\theta_k^{\eta\varepsilon}x^\varepsilon + x^{1/2-\varepsilon}/\sigma\| \leq 2/\rho|\mathcal{F}_k\},\\ &\leq \Pr\{W\theta_k^{\eta\varepsilon}x^\varepsilon \leq -x^{1/2-\varepsilon}/\sigma + 2/\rho|\mathcal{F}_k\},\\ &\leq \exp(-\frac{(-x^{1/2-\varepsilon}/\sigma + 2/\rho)^2}{2\theta_k^{2\eta\varepsilon}x^{2\varepsilon}}).\end{aligned} \tag{4.28}$$

for some $\delta > 0$.

□

Theorem 4.4.3. *(i)* $\lim_{k\to\infty}\sup_{M_k}\theta_k^{-2}\mathbb{E}[M^{-1+2\varepsilon}(X_k)|\mathcal{F}_k] \leq x$, *for* $x > \frac{4\rho^2}{\sigma^2}$.

(ii) $\lim_{k\to\infty}\inf_{M_k}\theta_k^{-2}\mathbb{E}[M^{-1+2\varepsilon}(X_k)|\mathcal{F}_k] \geq x$, *for* $x < \frac{\rho^2}{4\sigma^2}$.

Proof. To prove part (i), it suffices to just show that $\theta_k^{-2}M_k^{-1+2\varepsilon}$ is uniformly integrable, for $0 < \varepsilon < 1/2$.

Providing $1/6 < \varepsilon < 1/2$, for $x < (\sigma/2\rho)^\eta$, by Theorem 4.4.2, part (i) we have

$$\Pr\{\theta_k^\eta M_k \leq x|\mathcal{F}_k\} \leq \exp(-\theta_k^{-4\eta\varepsilon}(1/16\rho^2 - \frac{x^{1-2\varepsilon}}{4\sigma^2})^2|\mathcal{F}_k\} = o(\theta_k^2).$$

Therefore

$$\begin{aligned}\mathbb{E}[\theta_k^{-2}M_k^{-1+2\varepsilon}] &= \mathbb{E}[\theta_k^{-2}M_k^{-1+2\varepsilon}\mathcal{I}_{[\theta_k^\eta M_k \geq x]}] + \mathbb{E}[\theta_k^{-2}M_k^{-1+2\varepsilon}\mathcal{I}_{[\theta_k^\eta M_k < x]}]\\ &\leq x^{-1} + \theta_k^{-2}\Pr\{\theta_k^\eta M_k < x|\mathcal{F}_k\}\\ &= x^{-1} + o(1).\end{aligned}$$

To prove part (ii) we have

$$\begin{aligned}\mathbb{E}[M^{-1+2\varepsilon}(X_k)|\mathcal{F}_k] &= \int_{M^{-1+2\varepsilon}<\frac{\theta_k^2}{x^{1-2\varepsilon}}} M^{-1+2\varepsilon}(X_k) + \int_{M^{-1+2\varepsilon}\geq\frac{\theta_k^2}{x^{1-2\varepsilon}}} M^{-1+2\varepsilon}(X_k),\\ &\geq 0 + \theta_k^2/x^{1-2\varepsilon}\Pr\{\theta_k^{\eta}M_k \leq x|\mathcal{F}_k\},\\ &\geq \theta_k^2/x^{1-2\varepsilon}(1-\exp(-\frac{(-x^{1/2-\varepsilon}/\sigma+2/\rho)^2}{2\theta_k^{2\eta\varepsilon}x^{2\varepsilon}})),\end{aligned}$$

for $x > 4^{1/1-2\varepsilon}(\sigma^2/\rho^2)^{1/1-2\varepsilon}$. Therefore,

$$\lim_{k\to\infty}\inf_{M_k}\theta_k^{-2}\mathbb{E}[M^{-1+2\varepsilon}(X_k)|\mathcal{F}_k] \geq x^{-1+2\varepsilon},\ x > (4\sigma^2/\rho^2)^{1/1-2\varepsilon}. \quad (4.29)$$

□

Given the probabilistic behavior of the sample size M_k provided in Theorems 4.4.2 and 4.4.3, we now explore the quality of the related sampled estimator within (SCSR). According to the results in Chapter 3, a well-performing recursion is the one that shows an "asymptotic balance" between the sampling error ($\mathbb{E}[\|\tilde{H}(M_k,X_k)-h(X_k)\|^2|\mathcal{F}_k]$) and the recursion error (θ_k^2). The next theorem exhibits an interplay between these two types of error for (SCSR) augmented with (4.19).

Theorem 4.4.4. *As $k\to\infty$, we have $\theta_k^{-2}\mathbb{E}[\|\tilde{H}(M_k,X_k)-h(X_k)\|^2|\mathcal{F}_k] = \Psi(1)$.*

Proof. Write $S_m(X_k) = m\tilde{H}(m,X_k)$. Then $S_m(X_k)$ is a random walk with drift θ_k. Then we can rewrite (4.19) as follows:

$$M_k := \inf\{m > \nu_k : 1/\rho\hat{\sigma}(m,X_k)m^{1/2+\varepsilon} \leq \|S_m(X_k)\||\mathcal{F}_k\}.$$

Given $\mathcal{F}_k$, the above stopped process has the moving boundary $Q_m = \frac{\hat{\sigma}(m,X_k)}{\rho}m^{1/2+\varepsilon}$. Providing that as $m\to\infty$, $\frac{\hat{\sigma}(m,X_k)}{\rho} \overset{\text{wp1}}{\to} \sigma/\rho$, $Q(m)-Q(m-1) = O(\sqrt{m})$, letting

$$R(M_k,X_k) := \|S_M(X_k)\| - 1/\rho\hat{\sigma}(M_k,X_k)M_k^{1/2+\varepsilon}, \quad (4.30)$$

by (2.4) in [3] we have

$$R(M_k,X_k) \leq Y_{M_k} + 1/\rho(Q_{M_k} - Q_{M_k-1}). \quad (4.31)$$

Rewrite (4.30) as

$$\|\tilde{H}(M_k,X_k)\| = 1/\rho\hat{\sigma}(M_k,X_k)M_k^{-1/2+\varepsilon} + R(M_k,X_k)/M_k.$$

Therefore

$$\begin{aligned}\mathbb{E}[\tilde{H}^2(M_k,X_k)|\mathcal{F}_k] &\leq 2/\rho^2\mathbb{E}[\hat{\sigma}^2(M_k,X_k)M_k^{-1+2\varepsilon}|\mathcal{F}_k]+2\mathbb{E}[R^2(M_k,X_k)/M_k^2|\mathcal{F}_k]\\ &\quad +2/\rho\mathbb{E}[R(M_k,X_k)\hat{\sigma}(M_k,X_k)M_k^{-3/2+\varepsilon}|\mathcal{F}_k]. \qquad (4.32)\end{aligned}$$

First let's derive an upperbound for $\mathbb{E}[R^2(M_k,X_k)/M_k^2(X_k)|\mathcal{F}_k]$. By (4.31), We have

$$\begin{aligned}\mathbb{E}[\frac{R^2(M_k,X_k)}{M_k^2}] &= \mathbb{E}[\mathbb{E}[\frac{R^2(M_k,X_k)}{M_k^2}|M_k]],\\ &= \mathbb{E}[\frac{1}{M_k^2}\mathbb{E}[R^2(M_k,X_k)|M_k]],\\ &\leq 4\mathbb{E}[\frac{1}{M_k^2}(M_k+\mathbb{E}[Y_{M_k}^2|M_k])],\\ &= \mathbb{E}[O(M_k^{-1})]. \qquad (4.33)\end{aligned}$$

Also,

$$\begin{aligned}\mathbb{E}[R(M_k,X_k)\hat{\sigma}(M_k,X_k)M_k^{-3/2+\varepsilon}] &= \mathbb{E}[\mathbb{E}[R(M_k,X_k)\hat{\sigma}(M_k,X_k)M_k^{-3/2+\varepsilon}|M_k]],\\ &= \mathbb{E}[M_k^{-3/2+\varepsilon}\mathbb{E}[R(M_k,X_k)\hat{\sigma}(M_k,X_k)|M_k]],\\ &\leq \mathbb{E}[M_k^{-3/2+\varepsilon}\mathbb{E}[\hat{\sigma}(M_k,X_k)(Y_{M_k}\\ &\quad +1/\rho(Q_{M_k}f(M_k)-Q_{M_k-1}f(M_k-1)))|M_k]],\\ &= \mathbb{E}[M_k^{-3/2+\varepsilon}(\mathbb{E}[\hat{\sigma}(M_k,X_k)Y_{M_k}|M_k]\\ &\quad +1/\rho\mathbb{E}[Q_{M_k}f(M_k)-Q_{M_k-1}f(M_k-1)|M_k])],\\ &= \mathbb{E}[M_k^{-3/2+\varepsilon}(o(1)+\mathbb{E}[O(\sqrt{M_k})|M_k])],\\ &= \mathbb{E}[O(M_k^{-1+\varepsilon})]. \qquad (4.34)\end{aligned}$$

Given $\mathcal{F}_k$, $M_k \overset{\text{wp1}}{\to} \infty$, by Theorem 1 in [13], $\hat{\sigma}(M_k,X_k) \overset{\text{wp1}}{\to} \sigma$. Therefore, from (4.32), for k large enough

$$\mathbb{E}[\tilde{H}^2(M_k,X_k)|\mathcal{F}_k] = \Psi(\sigma^2/\rho^2\mathbb{E}[M_k^{-1+2\varepsilon}]). \qquad (4.35)$$

Then by Theorem 4.4.3 , $\mathbb{E}[h^{-2}(X_k)\tilde{H}^2(M_k,X_k)|\mathcal{F}_k] = \Psi(1)$; $\mathbb{E}[\frac{\|\tilde{H}(M_k,X_k)\|}{\|h(X_k)\|}|\mathcal{F}_k] = O(1)$, and hence:

$$\mathbb{E}[\|\tilde{H}(M_k,X_k)-h(X_k)\|^2/\|h(X_k)\|^2|\mathcal{F}_k] = \Psi(1).$$

□

The results provided so far in this section, present details on the behavior of the stopped process, as the baseline for proving efficiency of the recursion. The next theorem demonstrates asymptotic efficiency of (SCSR) under geometric/sub-geometric assumptions on the minimal rate of sampling.

Theorem 4.4.5. *Suppose in Theorem 4.4.4,* $\mathbb{E}[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2 / \|h(X_k)\|^2 | \mathcal{F}_k] \to \xi$, *for* ξ *being a finite positive constant.*

(R1) Let $\{\nu_k\}$ *be a geometric rate function with constant* $(1 - \frac{2l_0}{\beta} + \frac{(1+\xi)l_1^2}{\beta^2})^{-1}$. *Then the stochastic recursion with iterations defined by (SCSR) ensures Definition 4.2.1.*

(R2) Let $\{\nu_k\}$ *ensures Definition 1.4.3. Then the stochastic recursion with iterations defined by (SCSR), guarantees Definition 1.4.1.*

Proof. To prove (R1), given $\mathcal{F}_k$ by Assumptions (A2) and (A3), we have

$$l_0^2 z_k^2 \leq \|h(X_k)\|^2 \leq l_1^2 z_k^2. \tag{4.36}$$

Also, by Assumprions (A1), (A2), (A3) and (4.2.2), by [25] we have

$$\frac{\lambda}{2\beta^2}\|h(X_k)\|^2 \leq \|f(X_k) - f(x^*)\| \leq \frac{1}{\lambda}\|h(X_k)\|^2. \tag{4.37}$$

Therefore 4.2.1 follows by (4.22), corollary 4.3 of [25], (4.36) and (4.37).

In order to prove (R2), by Theorem 4.4.4, (A3), and by (4.21), there exists k_0, and $n_0 := \max\{k_0, K\}$ such that for all $k > n_0$,

$$\begin{aligned}
\mathbb{E}_\Omega[Z_{k+1}^2 | \mathcal{F}_k] &\leq (1 - \frac{2l_0}{\beta} + \frac{l_1^2}{\beta^2})Z_k^2 + \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2 | \mathcal{F}_k], \\
&\leq (1 - \frac{2l_0}{\beta} + \frac{l_1^2}{\beta^2})Z_k^2 + \frac{\xi l_1^2}{\beta^2} Z_k^2, \\
&= a Z_k^2,
\end{aligned} \tag{4.38}$$

where $\alpha := 1 - \frac{2l_0}{\beta} + \frac{(1+\xi)l_1^2}{\beta^2}$, and by $\beta > \frac{(1+\xi)l_1^2}{2l_0}$, $0 < \alpha < 1$. Hence letting $a_k := \mathbb{E}\|h(X_k)\|^2$ and $e_k := \mathbb{E}[Z_k^2]$, $q_k := \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2]$ and $d_k := e_1(1 - \frac{2l_0}{\beta})^k + \sum_{i=2}^{k-1}(1 - \frac{2l_0}{\beta})^{k-i} q_i + q_k$, for all $k \geq 1$ we have

$$e_k \leq \eta \alpha^k, \quad \eta = \max(e_{n_0}\alpha^{1-n_0}, \max_{1 \leq k \leq n_0}\{\alpha^{-k} d_k\}). \tag{4.39}$$

Then 1.4.1 is guaranteed by (4.36), (4.37), and Lemma 4.4.3. □

4.4.2 Adaptive-SCSR with Biased Estimators

In this section we assume that the gradient estimator is a biased estimator, and we want to derive the rate of convergence of (SCSR) augmented with the corresponding sampling rule. First we consider the finite difference (FD) estimator.

Theorem 4.4.6. *Let $\{Y_1^f, \ldots, Y_m^f\}$ be iid observations with mean $\mathbb{E}[Y_1^f(\cdot)] = f(\cdot)$ and covariance matrix satisfying (4.6). Also let $\tilde{H}(M_k, X_k)$, be the FD gradient estimator:*

$$\tilde{H}(M_k, X_k) = \frac{\tilde{F}(M_k, X_k + c_k) - \tilde{F}(M_k, X_k - c_k)}{2c_k},$$

where $\tilde{F}(M_k, \cdot) = M_k^{-1} \sum_{i=1}^{M_k} Y_i^f(\cdot)$. Let

$$M_k = \inf\{m > \nu_k \ : \hat{\sigma}(m, X_k)/m^{1/2(1-\varepsilon)} \leq \rho\sqrt{\|\tilde{H}(m, X_k)\|^2 - 2b_k^2}|\mathcal{F}_k\}, \tag{4.40}$$

where $1/3 \leq \alpha < 1$, $\sum_{k=1}^{\infty} \nu_k^{-1+\alpha} < \infty$, $c_k^2 = \Psi(\nu_k^{-\alpha})$ and $\varepsilon \geq \alpha/2$. Then the stochastic recursion (SCSR), augmented with (4.40),

(R1) is convergent to x^* almost surely;

(R2) ensures Definition (1.4.1), with $\varepsilon \geq 1/3$.

Proof. By (4.21), substitude $h(X_k)$, by $h(X_k) + b_k$, we have

$$\mathbb{E}_\Omega[Z_{k+1}^2|\mathcal{F}_k] \ \leq \ (1 - \frac{2l_0}{\beta} + \frac{l_1^2}{\beta^2})Z_k^2 + \frac{1}{\beta^2}\mathbb{E}_\Omega[\|\tilde{H}(M_k, X_k) - h(X_k) - b_k\|^2|\mathcal{F}_k],$$

and

$$\begin{aligned}\mathbb{E}[\|\tilde{H}(M_k, X_k) - h(X_k) - b_k\|^2|\mathcal{F}_k] &= \mathbb{E}[\|\Delta_{M_k}^+ - \Delta_{M_k}^-\|^2|\mathcal{F}_k] \\ &\leq 2(\mathbb{E}[(\Delta_{M_k}^+)^2|\mathcal{F}_k] + \mathbb{E}[(\Delta_{M_k}^-)^2|\mathcal{F}_k]).\end{aligned}$$

Moreover,

$$\begin{aligned}\mathbb{E}[4c_k^2(\Delta_{M_k}^+)^2|\mathcal{F}_k] &= \mathbb{E}[\|\tilde{F}(M_k, X_k + c_k) - f(X_k + c_k)\|^2|\mathcal{F}_k], \\ &= \mathbb{E}[M_k^{-1}(M_k\|\tilde{F}(M_k, X_k + c_k) - f(X_k + c_k)\|^2)|\mathcal{F}_k], \\ &= O(\nu_k^{-1}).\end{aligned}$$

Similarly $\mathbb{E}[4c_k^2(\Delta_{M_k}^+)^2|\mathcal{F}_k] = O(\nu_k^{-1})$. Hence by assumptions on $\{\nu_k\}$ and $\{c_k\}$,

$$\nu_k^{1-\varepsilon}\mathbb{E}[\|\tilde{H}(M_k, X_k) - h(X_k)\|^2|\mathcal{F}_k] \ = \ O(\nu_k^{-\varepsilon}c_k^{-2}) = O(1)$$

Thus by 4.21, Lemma 2 in [118], and $\sum_k \nu_k^{-1+\varepsilon} < \infty$, the stochastic recursion (SCSR) is convergent to x^* with probability one.

In order to prove (R2), we have:

$$\begin{aligned}\|\tilde{H}(M_k, X_k) - h(X_k)\| &= \|\frac{\tilde{F}(M_k, X_k + c_k) - f(X_k + c_k)}{2c_k} \\ &\quad - \frac{\tilde{F}(M_k, X_k - c_k) - f(X_k - c_k)}{2c_k} \\ &\quad + \frac{f(X_k - c_k) - f(X_k + c_k)}{2c_k} - h(X_k)\|\end{aligned}$$

Let

$$\begin{aligned}\Delta^+_{M_k} &:= \frac{\tilde{F}(M_k, X_k + c_k) - f(X_k + c_k)}{2c_k}, \\ \Delta^-_{M_k} &:= \frac{\tilde{F}(M_k, X_k - c_k) - f(X_k - c_k)}{2c_k}, \\ b_k &:= \frac{f(X_k - c_k) - f(X_k + c_k)}{2c_k} - h(X_k).\end{aligned}$$

By Spall, given $\mathcal{F}_k$, $b_k = O(c_k^2)$.

Providing that $b_k = \mathbb{E}[\tilde{H}(M_k, X_k)|\mathcal{F}_k] - h(X_k)$, and given $\mathcal{F}_k$, $\theta_k := h(X_k)$, we have

$$\begin{aligned}\|\tilde{H}(M_k, X_k) - \theta_k\| &= \|\Delta^+_{M_k} - \Delta^-_{M_k} + \mathbb{E}[\tilde{H}(M_k, X_k)|\mathcal{F}_k] - \theta_k\|; \\ &= \|\Delta^+_{M_k} - \Delta^-_{M_k} + b_k\| = \|\bar{Z}(M_k, X_k) + b_k\|; \qquad (4.41) \\ \text{or } \|\tilde{H}(M_k, X_k)\| &= \|\tilde{H}(M_k, X_k) - \mathbb{E}[\tilde{H}(M_k, X_k)|\mathcal{F}_k] + b_k + \theta_k\|, \\ &= \|\bar{Z}(M_k, X_k) + b_k + \theta_k\|, \qquad (4.42)\end{aligned}$$

where $\bar{Z}(M_k, X_k)$ ensures Lindeberg-Levy conditions.

We have

$$\begin{aligned}
\Pr\{M_k \le \theta_k^{-\eta} x | \mathcal{F}_k\} &= \Pr\{\sup_{m\in T(\theta_k)} \frac{\|\bar{Z}(m,X_k)+\theta_k+b_k\|^2 - 2b_k^2}{\hat{\sigma}^2(m,X_k)m^{2\varepsilon-1}} \ge \rho^2 | \mathcal{F}_k\}, \\
&\le \Pr\{\sup_{m\in T(\theta_k)} \frac{2\|\bar{Z}(m,X_k)+\theta_k\|^2}{\hat{\sigma}^2(m,X_k)m^{2\varepsilon-1}} \ge \rho^2 | \mathcal{F}_k\}, \\
&\le \Pr\{\sup_{m\in T(\theta_k)} \frac{\|\bar{Z}(m,X_k)\|^2+\theta_k^2}{\hat{\sigma}^2(m,X_k)m^{2\varepsilon-1}} \ge \rho^2/4 | \mathcal{F}_k\}, \\
&\le \Pr\{\sup_{m\in T(\theta_k)} (\frac{m^{1-2\varepsilon}\bar{Z}^2(m,X_k)}{\sigma^2-\varepsilon_1} + \frac{m^{1-2\varepsilon}\theta_k^2}{(\sigma^2-\varepsilon_1)}) \ge \rho^2/4 | \mathcal{F}_k\}, \\
&\le \Pr\{\sup_{m\in T(\theta_k)} (\frac{m^{1-2\varepsilon}\bar{Z}^2(m,X_k)}{\sigma^2} + \frac{m^{1-2\varepsilon}\theta_k^2}{\sigma^2}) \ge \rho^2/8 | \mathcal{F}_k\},
\end{aligned}$$

and the rest of the proof for Theorems 4.4.2 and 4.4.3, follows as before, just with a scaled upper bound on the value for x.

Now in order to prove Theorem 4.4.4, write $S_m(X_k) = 2c_k m\tilde{H}(m,X_k)$. Then $S_m(X_k)$ is a random walk with drift $2c_k(b_k+\theta_k)$. Rewrite (4.40) as follows:

$$M_k := \inf\{m > \nu_k : 2mc_k\sqrt{\frac{\hat{\sigma}^2(m,X_k)}{\rho^2 m^{1-2\varepsilon}} + 2b_k^2} \le \|S_m(X_k)\| | \mathcal{F}_k\}.$$

Given $\mathcal{F}_k$, the above stopped process has the moving boundary

$$Q_m = 2mc_k\sqrt{\frac{\hat{\sigma}^2(m,X_k)}{\rho^2 m^{1-2\varepsilon}} + 2b_k^2}.$$

Providing that as $m\to\infty$, $\hat{\sigma}^2(m,X_k) \overset{\text{wp1}}{\to} \sigma^2$. we prove $Q(m) - Q(m-1) = O(\sqrt{m})$. Letting $P_m := \frac{1}{2c_k\sqrt{m}}[Q(m) - Q(m-1)]$ we have:

$$P_m = \frac{\frac{\hat{\sigma}^2(m,X_k)m^{1+2\varepsilon} - \hat{\sigma}^2(m-1,X_k)(m-1)^{1+2\varepsilon}}{\rho^2 m} - 2/m b_k^2 + 4b_k^2}{\sqrt{1/\rho^2\hat{\sigma}^2(m,X_k)m^{2\varepsilon} + 2mb_k^2} + \sqrt{1/\rho^2\hat{\sigma}^2(m-1,X_k)\frac{(m-1)^{1+2\varepsilon}}{m} + 2\frac{(m-1)^2}{m}b_k^2}}.$$

Since for any constant α, $\{m^\alpha\}$ is a slowly varying sequence, $m^\alpha - (m-1)^\alpha = \Psi(m^{\alpha-1})$. Therefore $\frac{m^\alpha-(m-1)^\alpha}{m} = \Psi(m^{\alpha-2})$. Set $\alpha = 1+2\varepsilon$. Because $\varepsilon < 1/2$, $\frac{m^{1+2\varepsilon}-(m-1)^{1+2\varepsilon}}{m} = \Psi(m^{2\varepsilon-1}) = o(1)$. This proves that $\frac{2c_k}{\sqrt{m}}[Q(m)-Q(m-1)] = o(1)$. Now letting

$$R(M_k, X_k) := S_{M_k}(X_k) - Q_{M_k}, \tag{4.43}$$

by (2.4) in [3] we have

$$R(M_k, X_k) \leq Y_{M_k} + (Q_{M_k} - Q_{M_k-1}). \tag{4.44}$$

Rewrite (4.43) as (divide by $2M_k c_k$)

$$\begin{aligned} \|\tilde{H}(M_k, X_k)\| &= \frac{R(M_k, X_k)}{2M_k c_k} + \frac{Q_{M_k}}{2M_k c_k} \\ &= \frac{R(M_k, X_k)}{2M_k c_k} + \sqrt{\frac{\hat{\sigma}^2(M_k, X_k)}{\rho^2 M_k^{1-2\varepsilon}} + 2b_k^2} \end{aligned}$$

Therefore

$$\begin{aligned} \mathbb{E}[\tilde{H}^2(M_k, X_k)|\mathcal{F}_k] \leq\ & \mathbb{E}[\frac{\hat{\sigma}^2(M_k, X_k)}{\rho^2 M_k^{1-2\varepsilon}} + 2b_k^2|\mathcal{F}_k] + \mathbb{E}[\frac{R^2(M_k, X_k)}{2c_k^2} M_k^{-2}|\mathcal{F}_k] \\ & + \mathbb{E}[R(M_k, X_k)\frac{M_k^{-1}}{c_k}\sqrt{\frac{\hat{\sigma}^2(M_k, X_k)}{\rho^2 M_k^{1-2\varepsilon}} + 2b_k^2}|\mathcal{F}_k]. \end{aligned}$$

It can be seen that given $\mathcal{F}_k$,

(A) $\frac{\hat{\sigma}^2(M_k,X_k)}{\rho^2 M_k^{1-2\varepsilon}} = O_p(\nu_k^{-1+2\varepsilon})$;

(B) $b_k^2 = O(\nu_k^{-2\alpha})$;

(C) $\frac{R^2(M_k,X_k)}{2c_k^2} M_k^{-2} = O_p(\nu_k^{\alpha-1})$;

(D) $\frac{R^2(M_k,X_k)\hat{\sigma}^2(M_k,X_k)}{M_k^{3-2\varepsilon}c_k^2} = O_p(\nu_k^{\alpha-2+2\varepsilon})$;

(E) $\frac{2b_k^2 R^2(M_k,X_k)}{M_k^2 c_k^2} = O_p(\nu_k^{-1-\alpha})$.

Since $1/3 < \alpha < 1$ and $\varepsilon > \alpha/2$, $\mathbb{E}[\tilde{H}^2(M_k, X_k)|\mathcal{F}_k] = \Psi(\mathbb{E}[M_k^{-1+2\varepsilon}|\mathcal{F}_k])$; hence Theorem 4.4.4 and 4.4.5 follow accordingly for $\varepsilon \geq 1/3$.

□

The next theorem introduces a variant of the sequential sampling rule for generalized version of the biased estimator considered in Theorem 4.4.6. The effect of bias inherent in the estimator, yields degraded rate of convergence, as ε in Definition (1.4.1) is showed to be bounded away from zero with biased estimator. Theorem 4.4.7 shows that (SCSR) augmented with (4.45) ensures ε-efficiency for $\varepsilon \geq 1/3$.

Theorem 4.4.7. *Let $\{Y_1^f, \cdots, Y_m^f\}$ be iid observations with mean $\mathbb{E}[Y_1^f(\cdot)] = f(\cdot)$, and covariance matrix satisfying (4.6). Let $\tilde{H}(m, X_k)$, be a gradient estimator at X_k of the form*

$$\tilde{H}(m, X_k) = \bar{Z}(m, X_k) + b_k + \theta_k.$$

Suppose for $1/3 \leq \alpha < 1$, $\{\nu_k\}$ is a positive valued sequence with $\sum_{k=1}^{\infty} \nu_k^{-1+\alpha} < \infty$. Let $S_m(\cdot)$ be the iid sum of the form $S_m(X_k) = am\beta_k \tilde{H}(m, X_k)$, for a positive valued constant a, and sequence $\{\beta_k\}$ of the order $\beta_k = \Psi(\nu_k^{-\alpha/2})$.

For $\rho > 0$, and $\varepsilon \geq \alpha/2$, let

$$M_k = \inf\{m > \nu_k \ : \hat{\sigma}(m, X_k)/m^{1/2(1-\varepsilon)} \leq \rho \tilde{H}^*(m, X_k) | \mathcal{F}_k\}. \quad (4.45)$$

where $\tilde{H}^*(m, X_k) = \sqrt{\|\tilde{H}(m, X_k)\|^2 - 2b_k^2}$.

Providing $b_k = \Psi(\beta_k^2)$, the stochastic recursion (SCSR), augmented with (4.45),

(R1) is convergent to x^* almost surely;

(R2) ensures Definition (1.4.1), with $\varepsilon \geq 1/3$.

Proof. The claim follows by Theorem 4.4.6, replacing β_k with c_k and $a = 2$. □

4.5 Concluding Remarks

Our goal is to develop an adaptive sampling rule for use in parameter-free stochastic recursions to produce iterates that perform well in finite time while enjoying provable asymptotic consistency and efficiency under mild restrictions. The main idea underlying our adaptive sampling proposal is to continue sampling at a point until there is enough probabilistic evidence that the subsequent iterate X_{k+1} is of a higher quality (in terms of objective function value) than the current iterate X_k. The corresponding sample size M_k will then be used in estimating the function h and its derivatives at the incumbent point. The sample size determining rules M_k are designed to provide the stochastic recursion the flexibility to adapt to the problem structure and exhibit good performance in both finite and infinite time. This is as opposed traditional algorithms like SAA and SA where the sample size growth follows a deterministic rule (e.g. geometric) as the algorithm searches through potential solutions in the search space.

Chapter 5

EEG Pattern Recognition

Fast digital computers in combination with the ability to observe ElectroEncephaloGraph (EEG) signals from the human brain have created the possibility of *neurosprosthetics* using Brain-Computer Interfaces (BCI). These "thought controlled" prosthetic devices [77, 26, 74, 116, 114, 61, 78, 61] hold great promise in the lives of the severely motor impaired, since they pave the way for "wearable devices" that can be commanded and controlled by the subject's brain in much the same way as a well-functioning human limb.

The broad idea of such technologies is to detect and read a severely handicapped patient's EEG signals when the patient intends to perform a motor task, and then translate (or classify) the signal into meaningful intent. The patient's intended task which could be one of a set of simple pre-assigned tasks such as moving the right leg up, or grasping a cup with the left hand, can then be performed by fitting a prosthetic device. The overarching "Operations Research" to make this technology feasible is the interpretation of the signals from the patient in a seamless, near-instantaneous, and accurate manner so as to improve the patient's experience by effectively mimicking the operation of a real arm or leg.

Remark 5.0.1. *Brain activity can be monitored by either invasive or non-invasive methods. Invasive methods such as single-neuron recording, involve the surgical implantation of electrodes into the patient's brain [53, 94, 109, 114], which imposes significant clinical risks and has limited stability [63]. Non-invasive BCI systems seem more practical for everyday situations since they rely on analyzing on-line electrical brain activity recorded as EEG signals. EEG signals are recorded using a multi-channel electrode cap located over the motor cortex contralateral region of the brain [69, 52, 116, 114, 61, 78, 39, 115].*

A wide variety of other applications of EEG signal classification exist [52, 116, 114,

61, 78, 39, 115], but our intent in this dissertation is limited to the specific context of translating (or classifying) EEG data from a motor impaired patient into a prespecifed finite set of tasks, for use in an online context such as the operation of a wearable neuroprosthetic device. The EEG classification problem we consider here has recently generated a lot of attention among researchers [17, 49, 110, 94] due to its obvious usefulness.

A number of statistical and machine learning [37, 73, 18, 62, 106, 119, 68] techniques have been brought to bear with varying degrees of success towards constructing EEG pattern classifiers. Of these, there appears to be emerging evidence that methods that use a flexible probability model, e.g., an appropriate learning model or a hidden Markov model [107], are most appropriate for the context in question [119, 68]. In such a framework, a parametric probability model that characterizes EEG data as a function of each of the motor movements is constructed as a first step. The constructed model is then used within a learning setting to obtain the *a posteriori* "predictive" probabilities of the various motor actions when a specific EEG signal is observed from the patient. The motor action that the patient is attempting to engage in is then predicted to be that having the highest *a posteriori* probability.

While a learning framework for prediction is attractive, the crucial challenge in being useful in an online context is the *parameter estimation* step, where model parameters that will yield good predictive probabilities of motor actions are identified in an initial potentially time consuming training phase. Parameter estimation is accomplished by solving an optimization problem where an appropriate objective function, e.g., the error in expected prediction expressed as a function of the model parameters, is minimized. Such optimization is usually not straightforward because the objective function is routinely an expectation that is not known in analytic form. Instead, the objective function value at any given set of parameters can only be estimated by sampling a chosen fraction of the EEG signals available during the training phase. Using the complete set of training data as in [104], while helping to estimate the objective function effectively and thereby better solving the underlying optimization problem, tends to be time consuming and unfit for use in an online setting. Using too little EEG data as in [69, 67] has the diametric opposite effect — it results in fast parameter learning but understandably yields suboptimal parameters and consequently poor prediction rates.

Accordingly, our primary contribution in this Chapter should be seen as answering the question of how much to sample during optimization for parameter estimation within the context of constructing classifiers. Our insights lead to implementation of a Adaptive-SCSR proposed in Chapter 4, composed with a line search procedure, in order to decide how much EEG data to use when estimating the objective

function at a potential solution. As noted in Chapter 4, the sequential sampling strategy estimates and trades-off the bias and variance of the objective function's gradient estimate in order to obtain a probabilistic sense of whether further sampling of the objective function would actually be beneficial in terms of improving the quality of the subsequent iterate in the search process. The result of incorporating such a sequential sampling strategy is a stochastic recursive algorithm, called Line-search Adaptive SCSR (LIAS), that only *sparsely* samples the EEG data as it traverses the search space.

Our secondary contribution relates to implementation and has been used in slightly differing forms by recent prominent authors on the subject. Specifically, we incorporate an elaborate but rapidly implemented preprocessing step that "cleans" the EEG data and provides a good initial solution for use within Adaptive-SCSR. The preprocessing step consists of passing the data through a set of spectral and spatial filtering steps, in an effort to improve the resolution of signals monitored by the BCI. The main part of the filtering process is to construct a projection matrix through Common Spatial Pattern (CSP) so as to map the high dimensional low resolution data, to low dimensional and highly informative EEG data. Subsequently, the reformed data goes through a slight variation of K-Means clustering step to provide a high-quality initial solution for LIAS.

We provide extensive numerical results on the performance of the proposed paradigm using nine real datasets. Our numerical experience is promising on two accounts. First, the implementation times of the proposed framework appear to be at least one to two orders of magnitude lower than competing algorithms that have comparable prediction rates. This is primarily due to the use of an adaptive sampling scheme within Adaptive-SCSR. Second, the proposed algorithmic framework shows correct prediction rates in the range 63 percent to 96 percent across the nine data sets, with mean and standard deviation 76 and 10.1 respectively. These rates appear to be comparable to the best prediction rates that have been published over the last five years.

In what follows, we first describe the data set and then formulate the learning model and the optimization framework in detail. We provide numerical experience in Section 5.4 followed by concluding remarks in Section 5.5.

5.1 Data Description

5.1.1 What is EEG?

The variation of the surface potential distribution on the human scalp reflects functional activities emerging from the underlying brain. This surface potential variation can be recorded by affixing an array of electrodes to the scalp and measuring the voltage between pairs of these electrodes. The resulting data is called EEG [108].

Due to the large amount of information received from each electrode the analysis for continuous EEG is complex. The EEG is typically described in terms of (i) rhythmic activity and (ii) transients. Rhythmic activities that are mostly described by different waves, like so many radio stations can be categorized by frequency of their emanations (frequency bands), and in some case by the shape of their waveforms [101]. These classifications are due to either a certain distribution over the scalp or a certain biological significance, resulted from a set of rhythmic activity within a certain frequency range. Five types are particularly important which are listed in Table 5.1, together with the required biological stimulus.

Table 5.1: EEG in Terms of Rhythmic Activity

Name	Frequency Band	Stimulus
Delta	0.5-4Hz	deep sleep, defects in the brain in awaking state.
Theta	4-8Hz	emotional stress, deep meditation.
Alpha	8-13Hz	relaxed awareness and inattention.
Beta	13-35Hz	active thinking, active attention.
Gamma	larger than 35Hz	consciousness.

Additionally some features of the EEG are transient rather than rhythmic. Shocked and sharp waves could represent seizure activity in individuals with epilepsy. Examples of normal transient features are vertex waves and sleep spindles during normal sleep.

In our study, we focused on Motor Imagery/MI-based BCIs, which use sensorimotor rhythms (SMRs), such as Beta rhythms. These rhythms can only be recorded on the scalp over the sensorimotor cortex area. In particular we center on contralateral manifestation of imaginary hand movements which is a well-known neurophysiological phenomenon, and the features can be recorded in the contralateral

hemisphere.

5.1.2 Data Acquisition

MI-based BCIs use SMR features that are generated directly from the sensorimotor cortex. A significant decrease in the power level of SMRs can be observed on the contralateral hemisphere during the unilateral imagination of hand movements [95], and be used for BCI control.

The SMR features in this study, is provided by the Department of Medical Informatics, University of Technology Graz [24], which has been the data source for several prominent studies in the field of BCI. An array of 25 electrodes that are affixed to the sensorimotor cortex area of the scalp of nine human subjects, by a wearable EEG recording cap [24, 65, 111]. For each subject, two sessions are recorded during two different days where each session includes 6 runs, comprising 48 MI task-related trials. At the beginning of each trial, an arrow pointing either to the left, right, down or up (corresponding to one of the four classes left hand, right hand, foot or tongue) appears on the screen. Accordingly the subject is required to carry out the required imagery tasks until the arrow disappears from the screen in four seconds. It is then followed by a short break of variable size. As mentioned in Section 5.1.1, for the purposes of the current numerical study, we focus on contralateral manifestation of imaginary hand movements, and consider extracting only two classes (or motor actions) corresponding to the movements of the right or left hand.

The time interval in which the subject is the most concentrated on the specific MI task is chosen for training data extraction. This provides the model with correct brain source signals to be discriminated in two different motor actions. Hence following [65], the data extraction time for all subjects and all trials, is chosen to start in half a second after the MI task initiates and last for two seconds (that is, the time interval 0.5 to 2.5s from the calibration data is used for training the classifiers.).

5.2 Classification Model

We now present an abstracted problem formulation for constructing a classifier that probabilistically predicts motor actions corresponding to EEG signals obtained from a subject. We assume that the subject intends to perform one of $u < \infty$ known motor tasks, during which time EEG signals are observed; let the random

d-dimensional vector $Z|L := (Z_1, Z_2, \ldots, Z_d)|L, Z_i \in \mathbb{R}^d$ denote the EEG signal resulting from the subject performing a motor action $L \in \{1, 2 \ldots, u\}$. Suppose m pairwise realizations $(z^1, \ell^1), (z^2, \ell^2), \ldots, (z^m, \ell^m)$ of $(Z, L) \in \mathbb{R}^d \times \{1, 2, \ldots, u\}$ are observed through experimentation. The objective is to construct a classifier characterized by the conditional probability $\Pr\{L = \ell | Z = z\}$, using the available realizations $(z^1, \ell^1), (z^2, \ell^2), \ldots, (z^m, \ell^m)$. The *classifier* is a family of discrete probability mass functions $\Pr\{L = \ell | Z = z\}, \ell \in \{1, 2, \ldots, u\}$ parametrized by the observed EEG signal Z. The implication of such a classifier can then be used to deduce a human subject's intended action by simply observing EEG signals; for example, given the observed EEG signal $Z = z$, the intended motor action could be $\arg\max\{\Pr\{L = \ell | Z = z\} : \ell \in \{1, 2, \ldots, u\}\}$.

Any reasonable metric such as the expected probability of correctly predicting a motor action (where the mathematical expectation is computed with respect to the EEG signal Z) can be used in evaluating the competing classifiers. The classifier we propose is characterized through two well-defined components: (i) a flexible parametric multivariate distribution that is capable of representing the EEG signal Z adequately; and (ii) decision rule that incorporates knowledge of a classification task. The decision rule yields an optimization problem that seeks the "best" parameters in (i), that is, parameter settings that minimize the expected deviation of the resulting model's predictions from the true motor action in a certain probabilistic sense. In what follows, we describe each of the two components in some detail.

5.2.1 Maximum Likelihood Model

The conditional random variable $Z|L = \ell$ is assumed to be represented as a Gaussian Mixture Model (GMM), and has the density

$$g_{Z|L=\ell}(z, \theta) = \sum_{j=1}^{r_\ell} g(z|\mu_j(\ell), \Sigma_j(\ell))\pi_j(\ell), \tag{5.1}$$

where $\pi_j(\ell)$ is the mixture probability associated with the jth cluster for the conditioned label ℓ; $\theta := \{\mu_j(\ell), \Sigma_j(\ell), \pi_j(\ell), j = 1, 2, \ldots, r_\ell; \ell = 1, 2, \ldots, u\}$, and

$$g(z|\mu_j(\ell), \Sigma_j(\ell)) = \frac{1}{(2\pi)^{d/2}|\Sigma_j(\ell)|^{1/2}} \exp\{-\frac{1}{2}(z - \mu_j(\ell))^T \Sigma_j^{-1}(\ell)(z - \mu_j(\ell))\}, \tag{5.2}$$

is the d-dimensional Gaussian density with mean $\mu_j(\ell)$ and positive-definite covariance matrix $\Sigma_j(\ell) \in \mathbb{M}(d, d)$.

We denote the marginal probability model for the label L by $\nu(\ell) := \Pr\{L = \ell\}$. Given m realizations, $\mathbf{z}_m := \{z^1, z^2, \ldots, z^m\}$, the log-likelihood function $f(\theta|\mathbf{z}_m)$ is given by

$$f(\theta|\mathbf{z}_m) = \sum_{i=1}^{m} \sum_{\ell=1}^{u} \mathbb{I}\{\ell^i = \ell\} \left[\log \left(\sum_{j=1}^{r_\ell} g(z^i|\mu_j(\ell), \Sigma_j(\ell))\pi_j(\ell) \right) + \log\left(\nu(\ell)\right) \right], \tag{5.3}$$

with

$$\sum_{j=1}^{r_\ell} \pi_j(\ell) = 1, \ell = 1, 2, \ldots, u; \quad \sum_{\ell=1}^{u} \nu(\ell) = 1. \tag{5.4}$$

The parameter estimation problem is then that of maximizing the log-likelihood function $f(\theta|\mathbf{z}_m)$, that is:

$$\begin{aligned} \max_{\theta} \quad & f(\theta|\mathbf{z}_m) \\ \text{subject to:} \quad & \sum_{j=1}^{r_\ell} \pi_j(\ell) = 1, \pi_j(\ell) \geq 0, \ell = 1, 2, \ldots, u. \end{aligned} \tag{5.5}$$

Denoting $\lambda := (\lambda_1, \lambda_2, \ldots, \lambda_\ell)$, the Lagrange form of the problem in (5.5) is:

$$\begin{aligned} \max_{\theta,\lambda} \quad & f(\theta|\mathbf{z}_m) + \lambda_\ell \left(\sum_{j=1}^{r_\ell} \pi_j(\ell) - 1 \right), \\ \text{subject to:} \quad & \pi_j(\ell) \geq 0, \ell = 1, 2, \ldots, u. \end{aligned} \tag{5.6}$$

A first-order critical point to the problem in (5.6) satisfies the following $q \times q$ system of equations, where the number of parameters $q = u + \sum_{\ell=1}^{u}(1 + d + \frac{d(d+1)}{2})r_\ell$.

$$\begin{aligned} \nabla_{\mu_j(\ell)} f(\theta|\mathbf{z}_m) &= 0, j = 1, 2, \ldots, r_\ell; \ell = 1, 2, \ldots, u \\ \nabla_{\Sigma_j(\ell)} f(\theta|\mathbf{z}_m) &= 0, j = 1, 2, \ldots, r_\ell; \ell = 1, 2, \ldots, u \\ \nabla_{\pi_j(\ell)} f(\theta|\mathbf{z}_m) + \lambda_\ell &= 0, j = 1, 2, \ldots, r_\ell; \ell = 1, 2, \ldots, u \\ \sum_{j=1}^{r_\ell} \pi_j(\ell) &= 1, \pi_j(\ell) \geq 0, \ell = 1, 2, \ldots, u. \end{aligned} \tag{5.7}$$

As we will see shortly, the solution methods we construct will not consider the system (5.7) in its most general form. Instead, various parameters appearing in (5.7) will be fixed heuristically, resulting in a lower-dimensional parameter estimation problem. Also, due to the specific form of the function $f(\theta|\mathbf{z}_m)$ considered in this

paper, solving (5.7) to get the maximum likelihood parameter estimate $\theta^*|\mathbf{z}_m$ can be accomplished only using suitable iterative techniques involving the computation of the first and second derivatives of $f(\theta|\mathbf{z}_m)$.

Accordingly, strictly assuming the widths of the Gaussians be the only varying set of parameters, and denoting $P(\ell) := \sum_{j=1}^{r_\ell} g(z^i|\mu_j(\ell)\pi_j(\ell), \Sigma_j(\ell))$, $G_j(\ell) := g(z^i|\mu_j(\ell), \Sigma_j(\ell))$, and $A_j(\ell) := \Sigma_j^{-1}(\ell)(z^i - \mu_j(\ell))$, we have

$$\begin{aligned} \nabla_{\mu_j(\ell)} G_j(\ell) &= G_j(\ell) A_j(\ell); \\ \nabla^2_{\mu_j(\ell)} G_j(\ell) &= G_j(\ell)(A_j(\ell)(A_j(\ell))^T - \Sigma_j(\ell))^{-1}), \end{aligned}$$

leading to the following first and the second (partial) derivatives:

$$\nabla_{\mu_j(\ell)} f(\theta|\mathbf{z}_m) = \sum_{i=1}^{m} \mathbb{I}\{\ell^i = \ell\} \left[\frac{\pi_j(\ell) \nabla_{\mu_j(\ell)} G_j(\ell)}{P(\ell)} \right]; \tag{5.8}$$

$$\nabla^2_{\mu_j(\ell)} f(\theta|\mathbf{z}_m) = \sum_{i=1}^{m} \mathbb{I}\{\ell^i = \ell\} \left[\frac{\pi_j(\ell) P(\ell) \nabla^2_{\mu_j(\ell)} G_j(\ell) - \pi_j(\ell) \nabla_{\mu_j(\ell)} G_j(\ell) \nabla_{\mu_j(\ell)} P(\ell)}{P^2(\ell)} \right].$$

Three points are worthy of mention.

(i) As noted, identifying $\theta^*|\mathbf{z}_m$ by solving the system of equations in (5.7) through appropriate recursion involves computing the partial derivatives of $f(\theta|\mathbf{z}_m)$. As can be seen through the expressions in (5.8), the computation of such partial derivatives can become burdensome when entire amount m of available data is used towards such computation. This issue is particularly relevant to the current context of interpreting EEG signals, where the available "training data" from a human subject tends to be enormous; estimation methods that accelerate convergence rates through a more judicious use of existing data are of value.

(ii) Given the solution

$$\theta^*|\mathbf{z}_m := \{\mu_j^*(\ell), \Sigma_j^*(\ell), \pi_j^*(\ell), j = 1, 2, \ldots, r_\ell; \ell = 1, 2, \ldots, u|\mathbf{z}_m\},$$

to the parameter estimation problem (5.5), the classification probabilities are calculated in a straightforward way using Bayes rule as:

$$\Pr\{L = \ell | Z = \mathbf{z}_m\} = \frac{g_{Z|L=\ell}(z, \theta^*|\mathbf{z}_m)\nu(\ell)}{\sum_{\ell'=1}^{u} g_{Z|L=\ell'}(z, \theta^*|\mathbf{z}_m)\nu(\ell')}, \tag{5.9}$$

where the density $g_{Z|L=\ell}(z, \theta^*|\mathbf{z}_m)$ is the Gaussian mixture given in (5.1). The label achieving the highest probability, that is, $\ell^* = \arg\max_\ell \Pr\{L = \ell | Z = \mathbf{z}_m\}$, is then the classifier's prediction of the unknown label L when presented with the observed signal Z.

(iii) The estimator $\theta^*|\mathbf{z}_m$ is consistent in the sense that, under appropriate conditions, it is well known that under the maximum likelihood criterion,

$$\theta^*|\mathbf{z}_m \overset{\text{wp1}}{\to} \theta^*,$$

where θ^* is the solution to the corresponding limiting system.

5.3 Algorithm Line-search Adaptive-SCSR (LIAS)

In this section, we outline the algorithm for solving the simulation optimization problem specified through (5.5). Recall that the objective function in (5.5) is an expectation of a loss function representing the error in prediction by a GMM with parameter θ, where the expectation is taken with respect to the joint distribution of the location of the electrodes Z and the corresponding response L. The decision parameter θ is assumed to reside in a known set Θ. Since the prediction model $\hat{p}(Z;\theta)$ is a GMM, the objective function $f(\theta)$ cannot be assumed to enjoy any convexity properties per se, warranting the need to construct methods that account for the existence multiple local minima.

The algorithm we use for solving the problem in (5.5) has three components: (C.1) A preprocessor for data dimension reduction; (C.2) LIAS for detection of first-order stationarity points; and (C.3) a premature termination and restart strategy for use alongside C.2. The component C.1, is described in Section 5.3.1, which is executed on the collected "raw" data towards obtaining a reduced dataset $\mathcal{D} := (z^1, \ell^1), (z^2, \ell^2), \ldots, (z^m, \ell^m)$. The component C.2 is then executed (from a suitably chosen initial guess) on the sample-path problem towards estimating a first-order critical point of the function $f(\cdot)$. The logic in component C.3 is used to decide if the iterates obtained through C.2 are sufficiently close to a first-order critical point, at which time the procedure in C.2 is terminated and restarted from a new starting point that is strategically determined by C.2. Function estimates obtained at each of the identified first-order critical point estimators are compared sequentially to maintain an incumbent solution. When C.2 terminates, the out put forms the design model for testing a BCI performance. The proposed paradigm is called "Line-search Adaptive-SCSR" (LIAS), whose components are elaborated in details throughout this section (Fig.5.1). The MATLAB source code for Adaptive-SCSR is listed in Appendix 6.2.

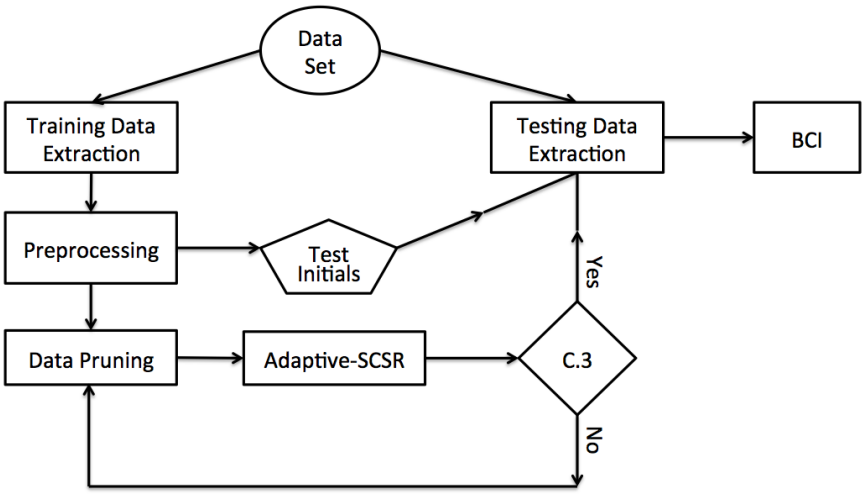

Figure 5.1: LIAS Paradigm : LIAS-classifier is proposed as an online learning tool for detecting EEG patterns.

5.3.1 Preprocessing in LIAS

EEG signal is one of the most complex biomedical signals due to its complex nature. Therefore, these types of signals are commonly pre-processed before the analysis procedure. Since pre-processing techniques affect the analysis results positively or negatively, the achievements of these pre-processing techniques are very important [89]. In signal processing, there are many instances in which an input signal to a system contains additional noise or unnecessary content which can reduce the quality of the desired portion. Consequently within BCI paradigms with the presence of this noise, the recognition performance can drop dramatically.

Accordingly our extensive numerical experience suggests that good finite-time performance of the classifier is dependent on two issues: (a) the removal of EEG data artifacts and consequent dimension reduction in the EEG signals; and (b) the availability of a good initial solution for the optimization step.

Towards ensuring (a), we subject the recorded EEG signals through a sequence of spectral and spatial filtering steps (a1 & a2), called as band-pass filtering and Common Spatial Pattern (CSP) respectively.

(a1) Spectral Filtering: Band-pass filter

We use a band power approach that involves extracting the power information from the signal for the SMRs. Finite Impulse Response (FIR) filters are one of the

main types of filters used in digital signal processing [2]. The FIR filter is used for signals to be band-pass filtered between 13 to 35 Hz .

(a2) Spatial Filtering: CSP

We perform spatial filtering on a set of measured EEG signals derived from multiple electrodes in order to

(i) select most informative and discard the irrelevant and redundant features; good subsets of features contain features that are highly correlated with the class and uncorrelated with each other;

(ii) extract localized information and to isolate the MI signals from the others; in other words, to improve the signal-to-noise ratio of the mu rhythm through extracting the task related EEG components, and eliminating the common components. Overall this provides signals that are easy-to-classify.

Well-known spatial filtering methods include the Laplacian, common average reference and CSP method [95]. Our interest in this study is CSP method, which has two set of filters in two class benchmarks. The first CSP filter amplifies the signal feature from the left-hand imaginary movement while the last CSP filter does the one from the right-hand imaginary movement.

CSP improves the signal-to-noise ratio of the EEG signals through localized information extraction, towards producing signals that are easy to classify. CSP designs a projection matrix whose columns are called CSP filters, aimed at extracting the key discriminative elements from the raw signal. CSP essentially takes an EEG signal which lives in high-dimensional space and projects it to a nearly orthogonal vector in a low-dimensional subspace that remains highly correlated the patient's motor actions. In all of our numerical experiments the original EEG signals were obtained from 25 "channels" resulting in a data vector that resides in 22 dimensions; CSP takes this raw data, projects the data into a data vector in six dimensions. In other words, the best potential spatial filters are naturally ranked in CSP; only the first three and last three spatial filters of the CSP are kept (or the first and last filter of each of the one-versus-rest CSP in the case of multi-class extension), thus selected six spatial filters and discarded the rest as irrelevant, redundant or not informative. The projection matrix produced out of training data, is part of the test initials in Fig. 5.1. Accordingly the same projection matrix is used to filter the test data.

In summary, CSP as a correlation-based feature selection method is performed in order to

(i) yield a large variance of signal for one class and a small variance of signal for the other class. This helps to design two spatial filters which led to the estimations of task-related source activities corresponding to the two tasks. The contrary effects of the filters on two classes contribute to estimating most discriminating sources and recovering some independent neurophysiological brain waves.

(ii) perform feature extraction, i.e. find fewer channels with the most information.

Remark 5.3.1. *On Curse of Dimensionality: When extracting relevant information from the channels, the more channels, the more information; but redundancy is an issue, because the more channels, the more correlations, and the more training data required for accurate classification. Small number of training instances, but highly dimensional raises the curse of dimensionality. It is generally accepted that the number of training instances should be at least ten times more than the features and that more complex classifiers require a larger ratio of sample size to features [4].*

(b) Initialization

Generating "invalid parameters" yields zero likelihood in (5.2) and degenerate results. Initial input patterns are required to describe the statistical properties of data to "some degree" of accuracy; otherwise extremely small values of the Gaussian densities in (5.2) result in occasional numerical errors such as underflow and division-by-zero in (5.8). Accordingly, this defines the parameter boundary for the likelihood function stated as (5.6), which needs to be avoided through the parameter estimation phase. If the input parameter lives close to the boundary of the parameter space and accordingly fails to explain a data point that is fed into classifier, due to the logarithmic scale, the overall loss in (5.6) will be infinity regardless of the likelihoods of the other points. Therefore any kind of randomized starting scheme would not facilitate proper initialization of the training model introduced in Section 5.2. As such, randomized starting schemes with no attempt to define the statistical properties of data, such as Lattice hypercube sampling [66] and randomized starting from the data set [41] would not mostly yield in successful initialization. Interestingly however, through the clustering approaches such as K-Means and Gonzalez [46], data dependent approximation of the parameters result in well-defined values for the likelihood function.

Remark 5.3.2. *K-Means clustering method divides the data extracted after preprocessing, into K clusters with K centroids, in a sequential manner when searching for the "nearest" centroid to each data point. The chosen distance measure*

is often the cosine measure, which has been shown to be effective when clustering EEG data [51, 65]. The resulting K modes together provide a high-quality initial solution for parameter identification in the current classifier problem.

While use of data-deriven clustering approaches, provides a successful initialization for the input candidates to the mixture model, it is observed that they often "dictate" immediate local solutions to the optimization problem. Therefore a local greedy method such as LIAS initiated with such approach, as will be observed shortly in Section 5.4, could stuck easily at the initial guess and never proceed any further into the search space. Accordingly our proposed heuristic for parameter initialization, is suggesting to perturb the initial guess by generating from a multivariate Gaussian with the same parameters as the clustering part. Numerically (as will be seen in Section 5.4.2) we have showed that this approach is taking us away from the local solutions suggested by K-Means clustering, while also providing "good" initial local patterns to our classifier.

Given this initial solution, the algorithm for training session of LIAS is listed in the next section.

5.3.2 LIAS Training: Algorithm Listing

As mentioned in Section 5.2 the decision parameter set is

$$\mathbf{x} := \{\mu_j(\ell), j = 1, 2, \ldots, r_\ell; \ell = 1, 2, \ldots, u\},$$

as , $\Sigma_j(\ell), \pi_j(\ell)$ are chosen to be fixed at the initial patterns. The stopping criteria, is to continue sampling at an iterate $X_k = \mathbf{x}$ (only) as long as the uncertainty in the estimated gradient is too high compared to the gradient estimate at the iterate (line 22 of Algorithm 1). When stop sampling, back-tracking line search [79] is used to decide on the step length of (SCSR), and calculate the next candidate solution X_{k+1} (line 53 and 54 of Algorithm 1). Under two conditions, line search back tracks: (i) if Armijo rule [79] is not satisfied (line 51 of Algorithm 1); (ii) if the resulted parameter is "invalid" (line 49 of Algorithm 1), which yields to zero likelihood values. And what we mean by "invalid parameter" is the same as discussed in the previous section.

SCSR is eventually terminated if reaching the maximum budget (line 63 of Algorithm 1), and returns the design model for the testing session. The algorithm listing for testing session is discussed in the next section.

Algorithm 1 LIAS Training

Given: Max Step length for Line Search s; Max number of iterations N_{max}; Initial Solution $\mathbf{x}_0$; coefficient of variation threshold c; sample variance perturbation ε, training data $Data := \{(z^1, \ell^1), (z^2, \ell^2), \ldots, (z^n, \ell^n)\}$; Line search constant c_1; Line search scaler β_l

Initialization

Set $k = 1$ ▷ initialize iteration number
Set $\hat{\sigma}(m, X_k) = \infty$. ▷ initialize standard error estimate
Set $\tilde{H}(m, X_k) = 0$. ▷ initialize gradient estimate
Set $m = 0$. ▷ initialize sample size
Set $\alpha_k = 0$. ▷ initialize step size
Set N_{max}.
Set $\nu = log(length(Data))/log(N_{max})$

Estimation

while further estimation is required **do**
    Set $Condition = 0$ ▷ initialize sampling stopping rule
    while $Condition = 0$ **do** ▷ keep sampling
        Set $m = m + 1$ ▷ update sample size
        if $m < length(Data)$ **then**
            Read the next datum (z^m, ℓ^m).
            Calculate instant function value ξ_m at X_k and (z^m, ℓ^m).
            Calculate instant gradient value Y_m at X_k and (z^m, ℓ^m).
            Update grand estimators

$$\tilde{F}(m, X_k) = \frac{m-1}{m}\tilde{F}(m, X_k) + \frac{1}{m}\xi_m(X_k).$$

$$\tilde{H}(m, X_k) = \frac{m-1}{m}\tilde{H}(m, X_k) + \frac{1}{m}Y_m(X_k).$$

        end if
        if $m > 1$ **then**
            if $m \leq length(Data)$ **then**
                Update variance estimator of the gradient estimate at X_k:

$$\hat{\sigma}^2(m, X_k) = \frac{m-2}{m-1}\hat{\sigma}^2(m-1, X_k) + \frac{1}{m}(Y_m(X_k) - \tilde{H}_{m-1}(X_k))^2.$$

            end if

if $c\frac{\hat{\sigma}(m,X_k)}{m^{1-\varepsilon}} \leq \|\tilde{H}(m,X_k)\|$ and $m > k^{\nu}$, or $m > length(Data)$ **then** ▷ sampling stopping rule.
  Set $M_k = m$.
  Set $Condition = 1$. ▷ quit sampling
  if $m > length(Data)$ **then**
    $s = \alpha_{k-1}$;
    $m = length(Data)$;
  end if
  $d_k = -m * \tilde{H}(m, X_k)$;

Line Search

Set $a_k = s$; ▷ Start Line Search with maximum step length
Set $response = 0$; ▷ Status initialization
Set $condition = 0$;
while $condition = 0$ **do**
  Set $X^+ = X_k + a_k d_k$;
  if $a_k < 1e-7$ **then**
    condition=1; ▷ Terminate line search due to the low step size.
    Set $X_{k+1} = X^+$;
    Set $\alpha_k = a_k$.
    Set $\tilde{F}(m, X_{k+1}) = \tilde{F}(m, X^+)$;
    Set $\tilde{H}(m, X_{k+1}) = \tilde{F}(m, X^+)$;
    Set $\hat{\sigma}(m, X_{k+1}) = \hat{\sigma}(m, X^+)$.
    if $response = 0$ **then**
      Report Error : Line search failed!
    end if
  else
    Update the grand estimator $\tilde{F}(m, X^+)$;
    Update the grand estimator $\tilde{H}(m, X^+)$;
    Update the grand estimator $\hat{\sigma}(m, X^+)$.
    Set $response = 1$ if observed no outlier/invalid parameter.
    if $response = 1$ **then**
      if $\tilde{F}(m, X^+) <= \tilde{F}(m, X_k) + c_1 a_k \tilde{H}(m, X_k)^T d_k$ **then**
        condition=1; ▷ Armijo rule
        Set $X_{k+1} = X^+$;
        Set $\alpha_k = a_k$;
        Set $\tilde{F}(m, X_{k+1}) = \tilde{F}(m, X^+)$;
        Set $\tilde{H}(m, X_{k+1}) = \tilde{H}(m, X^+)$;
        Set $\hat{\sigma}^2(m, X_{k+1}) = \hat{\sigma}^2(m, X^+)$;
      end if
    end if
  end if
  Set $a_k = \beta_l a_k$; ▷ tighten the search.
end while ▷ end of Line Search.

63: **if** $m = length(Data)$ and $k > N_{max}$ **then**
64: Terminate SCSR and return X_k; ▷ termination criterion.
65: **end if**
66: $k = k + 1$.
67: **end if**
68: **end if**
69: **end while**
70: **end while**

5.3.3 LIAS Testing : Algorithm Listing

Given the design model produced in training session and the projection matrix described in Section 5.1, the classifier is evaluated on testing data. In this session, the classifier labels each trial, and the fraction of correctly labeled trials defines the accuracy rate (line 34 of Algorithm 2). A pruning phase accompanies the labeling procedure for each trial, in order to only extract the data that provides discriminative information. To this end, given a probability $Threshold$, each data point is labeled as either responsive or not-responsive(unknown), based on the calculated posterior probability $\Pr(L|Z)$. If at a given point, the classifier fails to discriminate between the two classes, the data point is labeled as unknown, and does not participate to vote for labeling the corresponding trial (line 20 of Algorithm 2). Otherwise, the vote is entered according to the maximum posterior probability (line 12 of Algorithm 2). Eventually each trial is classified into one of the considered mental tasks, according to the majority vote (line 23 of Algorithm 2). Channel capacity is then calculated to infer the amount of true information that can be transferred to the interface (line 36 of Algorithm 2). Based on this metric, the performance of the classifier is heavily discussed in Section 5.4.

5.4 Numerical Experiments

We construct a computationally effective Gauss-Markov classifier capable of probabilistic prediction of EEG patterns. The input to classifier are the Graz EEG data features (see Section 5.1), and the output is the label for the test trial, while it can contain also confidence values (see Section 5.3). Like any statistical classifier, our proposed classifier consists of a probability model (outlines the inner structure for the classifier) and a decision rule or classifier type (to incorporate knowledge of a classification task), which are discussed fully in Section 5.2. The labels are then concluded based on the estimated classifier parameters that are adjusted within

Algorithm 2 LIAS Testing

Given: ℓ MI-Tasks, "Threshold" on posterior probabilities, last update X_k, testing data $(z^1, \ell^1), (z^2, \ell^2), \ldots, (z^n, \ell^n)$

Initialization

Set $t = 0$ ▷ initialize iteration number
Set $m = 0$ ▷ initialize observation number
Set Responses= 0 ▷ initialize responsive signal number
Set Unknown= 0 ▷ initialize unknown signal number
Set $True = 0$ ▷ initialize true classified trial number
Set $False = 0$ ▷ initialize false classified trial number

Estimation

while testing trials available **do**
    Set $t = t + 1$. ▷ adding a trial analysis.
    while in a single trial **do**
        $m = m + 1$.
        Read the next datum (z^m, ℓ^m).
        Set $[Probability, Class] = max_j(\Pr\{L = j | Z = z^m\})$.
        if $Probability > Threshold$ and $Class = 1$ **then**
            Set Responses=Responses+1.
            Set Class1=Class1+1.
        else if $Probability > Threshold$ and $Class = 2$ **then**
            Set Responses=Responses+1.
            Set Class2=Class2+1.
        else
            Set Unknown=Unknown+1.
        end if
    end while
    Set [MajorityVote(t), Decision(t)]=max(Class1,Class2).
    if MajorityVote(t)=0 **then**
        Report "unknown" label for trial t.
    else
        if Decision(t)=ℓ^m **then**
            Set $True = True + 1$.
        else
            Set $False = False + 1$.
        end if
    end if
end while
Calculate the error rate; $E_R = False/t$.
Calculate the probability of unknown responses; $P_U = Unknown/m$.
Calculate the Channel Capacity; $C_{Cap} = P_U(1 + E_R\log(E_R) + (1 - E_R)\log(1 - E_R))$.

the training session and minimizes the classification error of the training set in a supervised fashion.

Training and testing sets are chosen along the lines of [65] in two base cases: case 1 trains the classifier on the first session, and test it on a randomly selected half of the test data; case 2 trains the classifier on the first session, plus a randomly selected half of the second session, and tests it on the rest of the second session. We emphasize that the set of data for online testing was not used in any way when training the classifier in any of the base cases.

The parameter estimation within the classifier is perfumed through LIAS introduced in Section 5.3, as a global simulation optimization algorithm that solves for the optimal parameters through efficient sampling from our large dataset. The optimization problem is formed as a negation of (5.5) in Section 5.2. The algorithm demonstrates convergence, but more importantly, the prediction accuracy from implementation on the nine real datasets dominates the currently available best algorithm for the purpose.

The performance metrics to be considered throughout this section are the accuracy rate, channel capacity and data utilization. Accuracy rate represents the fraction of testing trials that are labeled correctly with each set of input patterns. Channel capacity is the information rate (in units of information per unit time) transmitted into the BCI interface, achieved with arbitrarily small error probability. Eventually by data utilization, we mean the amount of data that is utilized by the classifier in order to test the performance on the test set.

This section goes in details about the behavior of these metrics in different test environments. We start off this section by introducing the main numerical challenges faced while implementing LIAS on Graz EEG data set. Then we outline heuristics used to get around numerical issues. We warp up this section by performance evaluation of the proposed classifier and comparison to prominent competing methods.

5.4.1 Numerical Challenges

Initial Sampling

Through the theorem of Adaptive-SCSR outlined in Chapter 4, we proved that in order for almost sure convergence, we need a minimum rate for initial sampling sequence. Intuitively, the theorem states that the initial estimates of the trajectory of the search need to fulfill a minimum level of accuracy in order to lead the recursion toward the true solution. This fact is in common with most of the

literature in relative-width sequential sampling rules, where it is crucial to obtain high coverage probabilities based on the choice of the initial sample size. However, the choice of this parameter is often model-dependent [59], and hence requires human intervention to come up with a good choice. Adaptive-SCSR on the other hand, provides an appropriate initial accuracy and ensures convergence based on a general condition on the sub-geometric rate of initial sampling. In other words, at each iteration, the estimates should be accurate enough in order for Adaptive-SCSR to decide on the sample size based on the quality of the current iterate. Moreover, too noisy estimates would even make the iterates astray per chance and never converge. Therefore the initial sampling sequence or the so-called "escorting sequence", would impose the estimates to be accurate enough, while Adaptive-SCSR stopping rule prevents too much sampling in order for efficiency(Fig. 5.2).

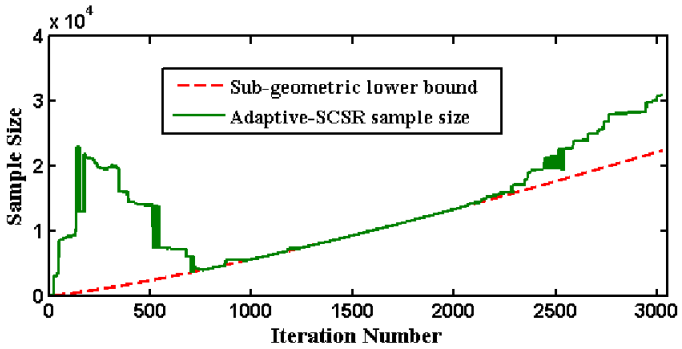

Figure 5.2: Sampling behavior of Adaptive-SCSR on Data Set 7 (DS7): Extremely high sample size within initial iterations reveals absorption of the SCSR iterates to the K-Means solution. Thanks to a fixed step size, Adaptive-SCSR owns the tracking capability to run away from the basin of attraction of K-means and approach a *better* local solution. The initial sampling sequence or the so-called "escorting sequence" facilitates convergence behavior accordingly, and Adaptive-SCSR stopping rule controls careful increase in the accuracy of the estimates when close to the solution.

Sampling Fluctuation

The idea behind sampling techniques is to be frequent with noisier estimates when far away from the local solutions, and be very accurate when close to the solutions. Even though the logic behind the proposed stopping rule imposes high rate of sampling when perceived vicinity of the true solution, sampling fluctuations often happens even when close to the true solution. Accordingly this behavior seems to

be independent of the quality of the parameter trajectory and still only responds to the iterative initial accuracy of the estimates, that is determined based on the polynomial index of the escorting sequence ν and the ε-value (Fig. 5.3).

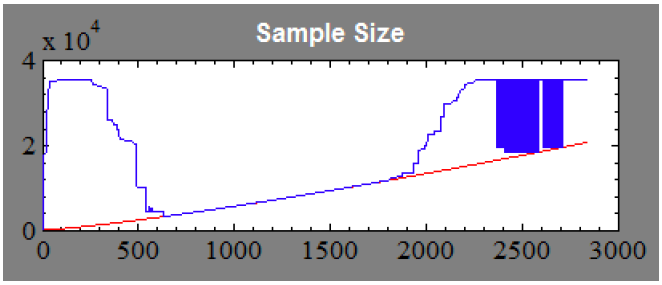

Figure 5.3: LIAS with fixed step size is performed on DS7. Sampling fluctuation is observed in the vicinity of local minima using a low ε-value.

In the case of slow initial sampling, the escorting sequence will eventually catch up with whatever large initial sample size required to eliminate fluctuation. The slowly growing escorting sequence takes the estimators to an accuracy level, that makes the iterates to leave the the state of noisy estimates (i.e. when the sample size lies at the initial sampling) completely, get rid of sampling fluctuations, and reach an increasing trend. Unfortunately however, this will cause more effort, than the case where we set a "correct rate" of increase for the escorting sequence ahead of time, due to the fluctuations that hit the maximum budget each time (Fig. 5.4).

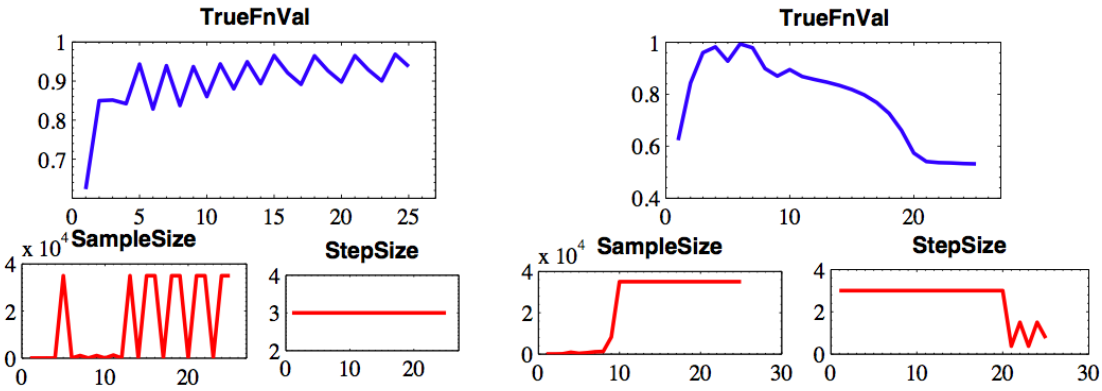

Figure 5.4: LIAS performance on DS5; $\nu = 1.25$ in the left panel, vs $\nu = 3.25$ in the right panel. Due to the low rate of escorting sequence, SCSR failed to exhibit convergence to a local solution in 25 iterations, where as on the right panel with higher rate of escorting, the true function value has dropped off to the vicinity of the local minima in less than 20 iterations. Moreover with higher rate of escorting, fluctuations are gone and the sequel has a smooth increasing trend.

To "obtain" a correct rate, we can gradually increase the rate based on the presence of fluctuations. However this may cause the algorithm work with some human intervention that is not appealing to adaptive schemes. Rather the ε increment may impose more sampling specially when we are close to the solution. Each time it increments the sample size a little more, and the estimates are reaching more accuracy, until it falls to the understanding of existence of a local min, and the sample size reaches the maximum level.

After all, in order to automatically avoid sampling fluctuation, and degrade the effect of the choice for ν and ε parameters, we impose trend condition so as to yield non-decreasing trend in the sample size. Since the iterates improve the search direction presumably, trend condition accelerate perceiving the vicinity of the local solutions by the SCSR iterates. Fig. 5.16 through 5.19 in Section 5.4.2 show the performance of LIAS under trend condition.

Choice of the free parameters

Behind the theory of Adaptive-SCSR discussed in Chapter 4, the choice of c-value as well as the learning rate $(1/\beta)$ appears to be more model-dependent, as they deem to be dependent on l_0 and l_1, and in the first look, it seems difficult to make a general recommendation concerning their choice; simply because the theoretical specifications (e.g. the values for l_0 and l_1) are unavailable. In such cases the choice for the parameters are either empirically based on previous knowledge on adaptive learning, or it is advisable initially to collect a small number of observations to obtain a "good" choice for the free parameters. For instance, as discussed in Chapter 2, the characteristics of a good learning rate has been long studied in the literature. However this is a pitfall in the context of adaptive learning, since we aim to define a parameter free method to avoid any kind of specification error. To this end, there are a few strategies that helped us automatize the parameter setting phase of the procedure in the practical setting:

(i) A general choice for the escorting sequence also drops the need for the c-value (appearing in Adaptive-SCSR stopping rule in Algorithm 5.3.2) being very close to zero; otherwise the sampling rule cannot suffice the minimum accuracy level for the early estimates and the iterates most likely astray per chance and convergence is violated. Under the weak rate conditions stated in the theory of Adaptive-SCSR in Chapter 4, convergence of the algorithm is supported almost surely, and c getting any value between zero and one, would suffice for a good performance.

Remark 5.4.1. *We note that in the context of sequential sampling, this*

constant determines the half width of the target relative confidence interval on the estimates of the true mean of the population [27]. We specifically choose it to be large value, and show that even with a pre-specified large width of the confidence interval, the sampling rule proposes enough sampling for a good improvement. We also note that if we had a fixed width sequential sampling rule, this size of the confidence interval may result in low sample sizes and delayed convergence. However the relative width confidence interval resulted by our procedure prevents getting stuck at a low level of accuracy.

(ii) Although the theory requires the knowledge of l_0 and l_1 (ineq. 4.36), to pick the value for step size, which are unavailable to us in practice, they can be estimated and be used within line search method. Specifically the upper bound on the step length, which is the initial solution for line search, needs to be chosen carefully, in order to avoid "stalling behavior" (Fig. 5.5 and 5.6).

Accordingly, as suggested by theory, we provide an estimate through the second order information available within the history of the search. This upper bound provides us with a well-working choice when implementing back-tracking line search (Fig. 5.7). Given the minimization problem under consideration, line search shrinks the initial guess down until a sufficient reduction condition on the function value is satisfied. Section 5.3 is more rigorously speaking out the steps for line search. Also we note that beside the maximum step length, there are other parameters within the line search, whose reference ranges are not problem-specific and the performance is not critically sensitive to their choices. Otherwise, the need to set yet another set of arbitrary parameters would not be appealing in the context of adaptive-learning.

(iii) Another core parameter in Adaptive-SCSR stopping rule is the ε-value (again, appearing in Adaptive-SCSR stopping rule in Algorithm 5.3.2) or the penalizing constant that reduces sampling variability specially when close to the true solution. The larger the ε-value, the closer BSG and Adaptive-SCSR, and the less the efficiency. On the other hand, choosing a value very close to zero cannot guarantee low enough sampling variability when close to the true solution, and therefore degraded performance in finite time (Fig. 5.8). So we preferably choose ε to be a "relatively" small value, e.g. 0.2 .

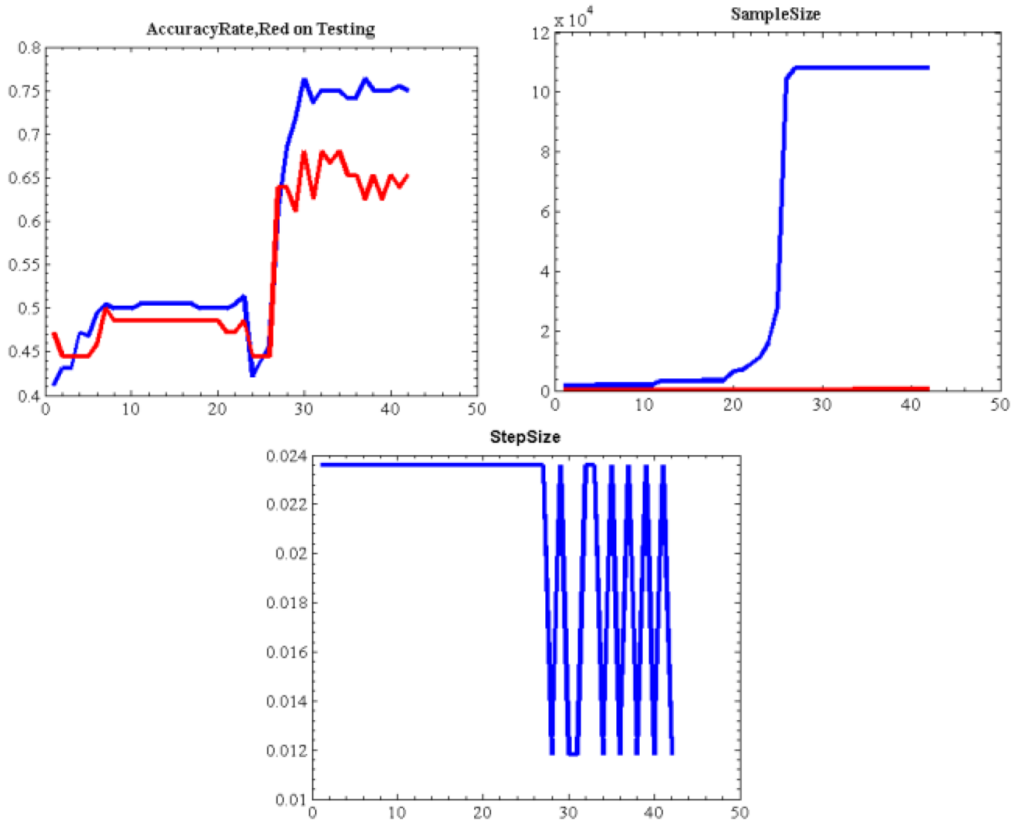

Figure 5.5: LIAS performance on DS4. Maximum step length in line search is chosen to be one, as an arbitrary value. In this case, (only by chance) it is shown to be large enough, as line search requesting a value below 0.024.

The issue of number of GMM clusters

One shortcoming of parametric approaches is that the number and nature of classes and components must be specified prior to estimation. Having to pick the choice for this parameter increases the likelihood of specification error, unless we are careful enough to choose this parameter. This factor is defining the complexity of the approximating distribution. Too small prevents the classifier from learning the sample distributions well enough (Fig. 5.9 and 5.10) and too large will result in smaller partitions leading to over-fitting and lower initialization accuracies for unseen data (Fig. 5.11 and 5.12).

More importantly, too large will definitely lead to singularities and degenerate results when the amount of training data becomes insufficient. Therefore we need

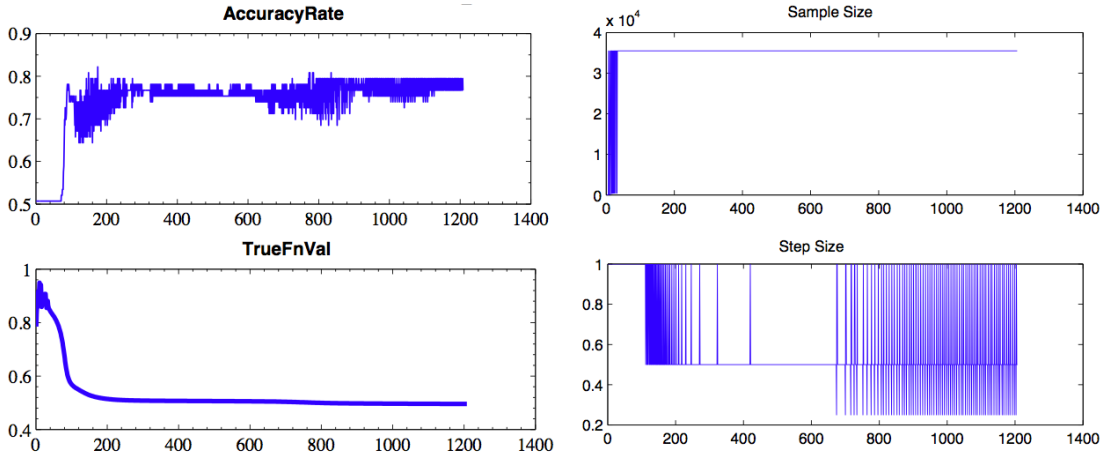

Figure 5.6: LIAS performance on DS7. Maximum step length in line search is chosen to be one, as an arbitrary value. Stalling behavior and degraded rate of convergence is observed, to the extent that it takes about 200 costly iterations of SCSR to observe a local solution.

to balance between fit versus generality.

Remark 5.4.2. *One way to avoid singular covariance when the number of clusters is too large (specially when the number of training instances are less than the number of decision variables), is to constrain the off-diagonal elements of the covariance matrix to zero. In this case, the platform fits multivariate normal distributions that have no correlations between the variables.*

Several model selection methods have been proposed to estimate the number of components of a mixture, either as a pre-fixed value, or being adjusted iteratively [42]:

- Bayesian Information Criterion (penalized likelihood approach) : The minimum entropy criterion is based on the argument that optimal clustering would maximize the information shared between the clustering and data. The minimum conditional entropy criterion can be used to find the optimal number of clusters. Despite its simplicity, BIC performs well in simulation studies.

  It has been shown that, by using Havrda Charvat structural entropy measure, the conditional entropy can be estimated without any assumption about the distribution of the data. However without information about the underlying probability distributions, estimating the conditional entropy is difficult. A solution is to use the Parzen window method for density estimation as suggested.

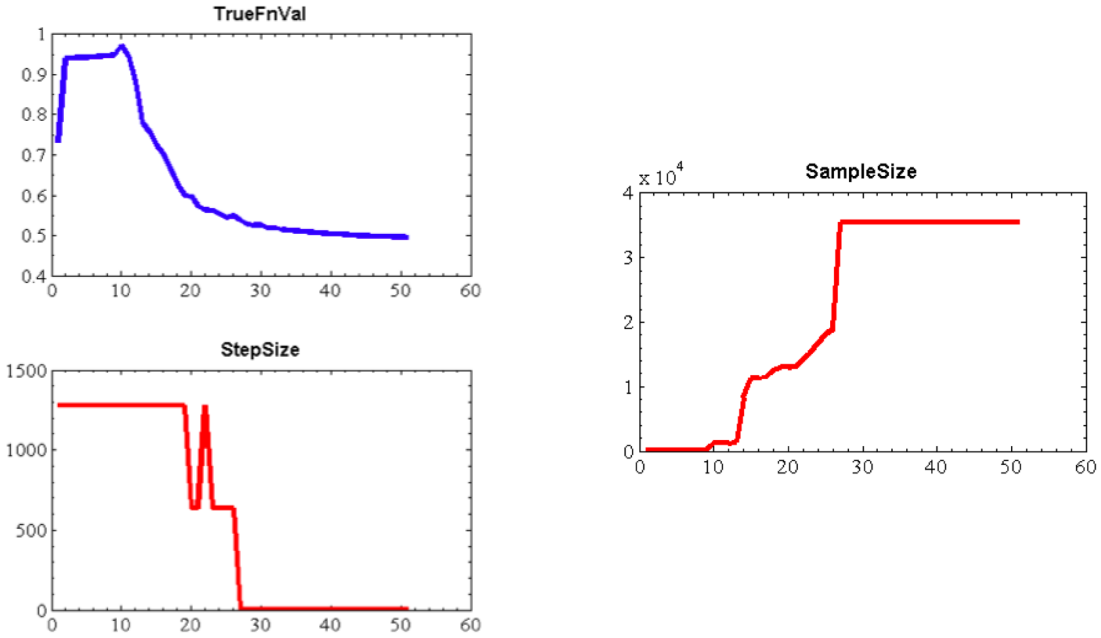

Figure 5.7: LIAS performance on DS7. Maximum step length in line search is chosen according to the second order information available through the history of SCSR search. It is shown that this choice is successfully giving enough "freedom" to line search, so as for dragging the iterates towards solution in big cheap steps, hence improving efficiency.

- Figueiredo-Jain (FJ) algorithm:

  Generally covariance update is not of interest because when one has insufficiently many points per mixture, estimating the covariance matrices becomes difficult, and the algorithm is known to diverge and find solutions with infinite likelihood, unless one regularizes the covariances artificially (by using soft constraints on the covariance matrix) or using annihilating technique suggested by Figueiredo-Jain that adjusts the number of components during estimation by annihilating components that are not supported by the data. Under the influence of the minimum-entropy prior that involves iterative tests for the existence of a particular subpopulation, this method starts with too many components all over the space (for instance by setting one component for every single training sample), and annihilates unnecessary ones.

  This method also avoids getting stuck at a local maxima of the likelihood, and finds a global solution. Local maxima of the likelihood arises when there are too many components in one region of the space, and too few in another,

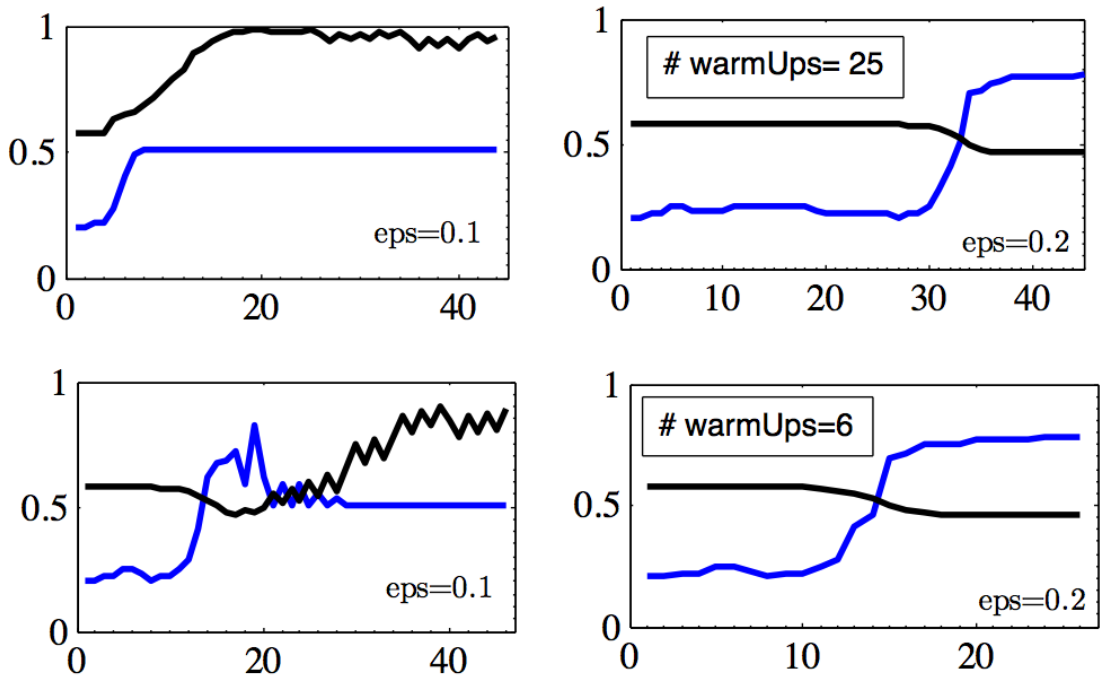

Figure 5.8: Two restarts of LIAS on DS7 is shown with different initializations; for each restart, we set $\epsilon = 0.1$ in the left panel, vs $\epsilon = 0.2$ in the right. Learning curve is in black, and the blue curve is the accuracy rate. Convergence is observed with $\epsilon = 0.2$, while for $\epsilon = 0.1$, the accuracy cannot go higher than a coin-toss percentage within 50 iterations.

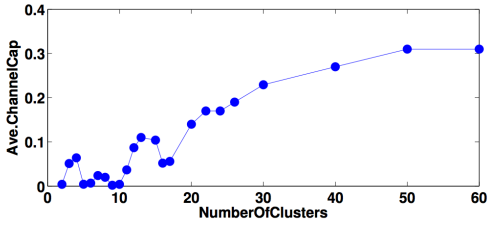

Figure 5.9: Channel Capacity of the classifier on DS7-training, across different number of clusters.

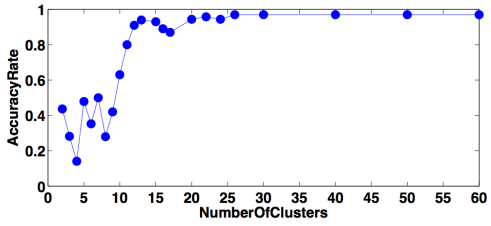

Figure 5.10: Accuracy Rate of the classifier on DS7-training, across different number of clusters.

because it is generally hard to move components across low- likelihood regions (like as in EM). By starting with too many components all over the space and gradual elimination of the ones that are becoming singular, local solutions can be avoided.

- Full Bayesian setting using Dirichlet Process priors. This approach requires

the use of the computationally expensive MCMC.

Overfitting Phenomenon

When the model works pretty well on the training set and does poor on the testing set, we say that it overfits. In other words, it overfits if the model cannot be generalized to the new data. So the output parameter from training set, might work very well on the training set, but does not well in the testing session. In such cases, while the learning curve has a steady decreasing tendency, the accuracy may stay the same or can go down occasionally. The more complicated the training model, the higher the chance to overfit. Fig. 5.11 and 5.12 show over fitting phenomenon on DS7 for number of clusters being greater than 10. [105] proposed using low iteration steps to deal with overfitting. Also adding a scaled unity matrix to the calculated covariance matrix reduces the risk of overfitting when the number of training samples is low in comparison to the number of clusters [9].

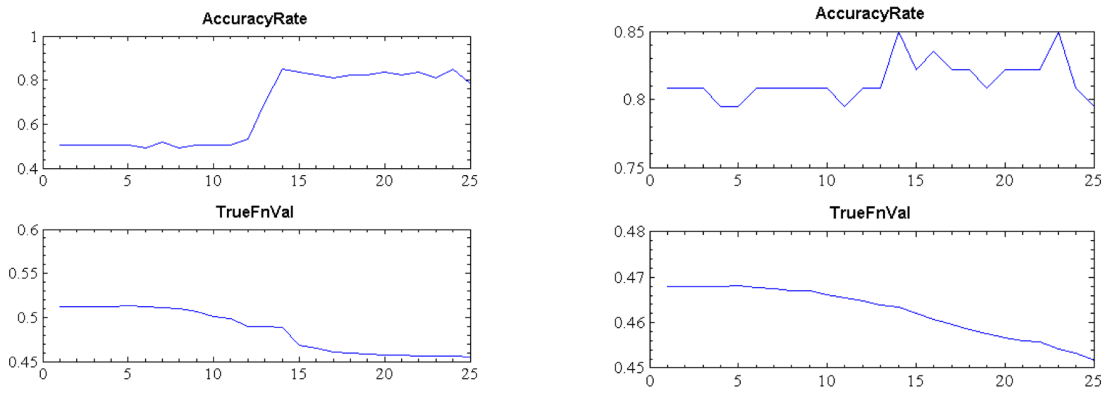

Figure 5.11: LIAS performance on DS7: number of Gaussian clusters equal to 9 on the left vs 12 on the right. Using a large number of clusters (greater than 10), although the learning curve has a decreasing tendency, the accuracy rate is not improving, due to the over-fitting phenomenon.

5.4.2 Heuristics for Implementation

In order to successfully implement the proposed learning paradigm, multiple heuristics are applied, which are broadly described in this section.

(H1) Choice of Initialization Strategy. Generating "invalid parameters" yields zero likelihood and degenerate results. Initial input patterns are required

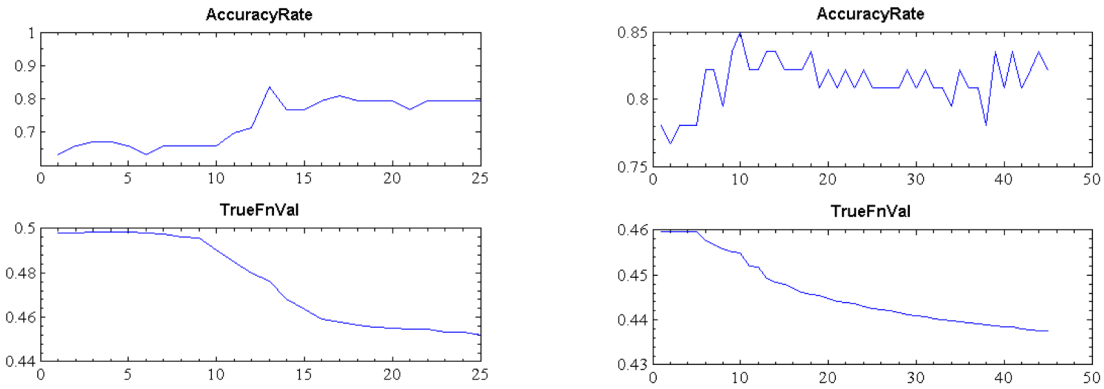

Figure 5.12: LIAS performance on DS7: number of Gaussian clusters equal to 10 on the left vs 13 on the right. Using a large number of clusters (greater than 10), although the learning curve has a decreasing tendency, the accuracy rate is not improving, due to the over-fitting phenomenon.

to describe the statistical properties of data to "some degree" of accuracy; otherwise extremely small values of the Gaussian densities in (5.2) result in occasional numerical errors such as underflow and division-by-zero. Accordingly, this defines the parameter boundary for the likelihood function stated as (5.7), which needs to be avoided through the parameter estimation phase. If the input parameter lives close to the boundary of the parameter space and accordingly fails to explain a data point that is fed into classifier, due to the logarithmic scale, the overall loss will be infinity regardless of the likelihoods of the other points. Therefore any kind of randomized starting scheme would not facilitate proper initialization of the training model introduced in Section 5.2. As such, randomized starting schemes with no attempt to define the statistical properties of data, such as Lattice hypercube sampling [66] and randomized starting from the data set [41] would not mostly yield in successful initialization. Interestingly however, through the clustering approaches such as K-Means and Gonzalez [46], data dependent approximation of the parameters result in well-defined values for the likelihood function.

While use of data-deriven clustering approaches, provides a successful initialization for the input candidates to the mixture model, it is observed that they often "dictate" immediate local solutions to the optimization problem. Therefore a local greedy method such as LIAS initiated with such approach, is observed to stuck at the initial guess and never proceed any further into the search space. Interestingly however, multiple restarts of K-Means on different data sets show that K-Means does not provide high quality solutions. Moreover, all the local solutions suggested by K-Means fall into a

small region (Fig. 5.13).

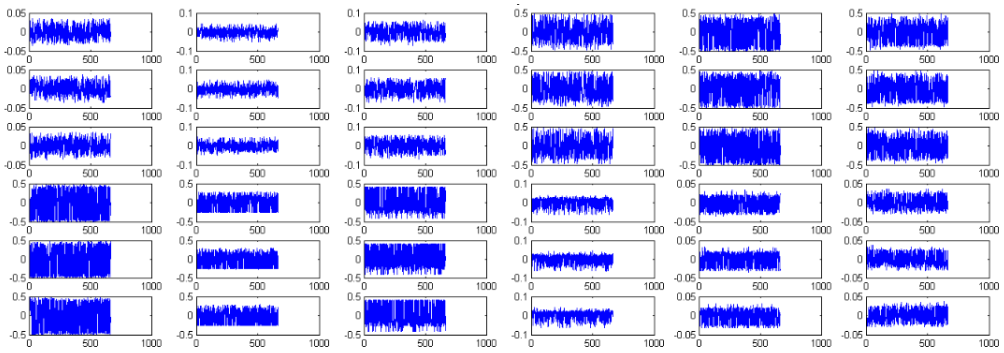

Figure 5.13: Parameters in 36 diminutional space, generated by multiple re-starts of K-Means clustering.

With this motivation, two heuristics are proposed for parameter initialization: (i) to initiate Line-Search Adaptive-SCSR with a few iterations with a fixed "large" step size- or so-called "Warm up" period; (ii) to perturb the initial guess by generating from a multivariate Gaussian with the same parameters as the clustering part. The goal in both approaches is to get away from the local solutions suggested by K-Means clustering, while also providing "good" initial local patterns to our classifier. The former approach showed to be working in the preliminary results (Fig. 5.14) with an arbitrary size of the warm up session. However a natural question arises as to how to choose the size for the warm up session. Fig. 5.15 shows the performance of Adaptve-SCSR with different number of warm-up iterations.

Although Fig. 5.15 shows that the goal is achieved even with small size of warm-up, we are still reluctant to leave this as a free parameter in favor of reaching a fully-adaptive procedure. Therefore we proceed to the next initialization idea. Motivated by the fact that all the K-Means solutions lie within a small region, just shaking this initial guesses off a little, would take us away from the basin of attraction of K-Means. Accordingly, we perturb the initial solutions suggested by K-Means, and use this new set, as the initial patterns for LIAS to start with. Fig. 5.16 through 5.16 show the performance of Adaptive-SCSR with "perturbed K-Means" initialization.

(H2) Addressing Outliers in the Data set.

Providing the aforementioned procedure brings valid initial parameters to the classifier, the presence of a few outliers within the training data set is inevitable. Fig. 5.20 shows the behavior of the method when the coming data injected into the model, are outliers. In order to avoid numerical errors due

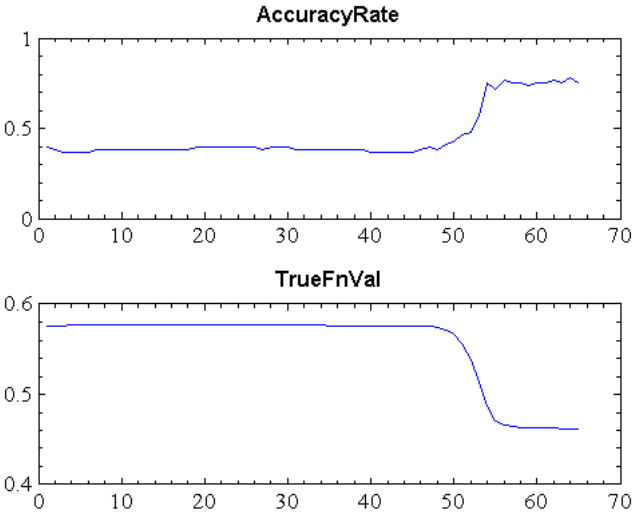

Figure 5.14: LIAS performance on DS7, initiated with 45 iterations of warm-up session.

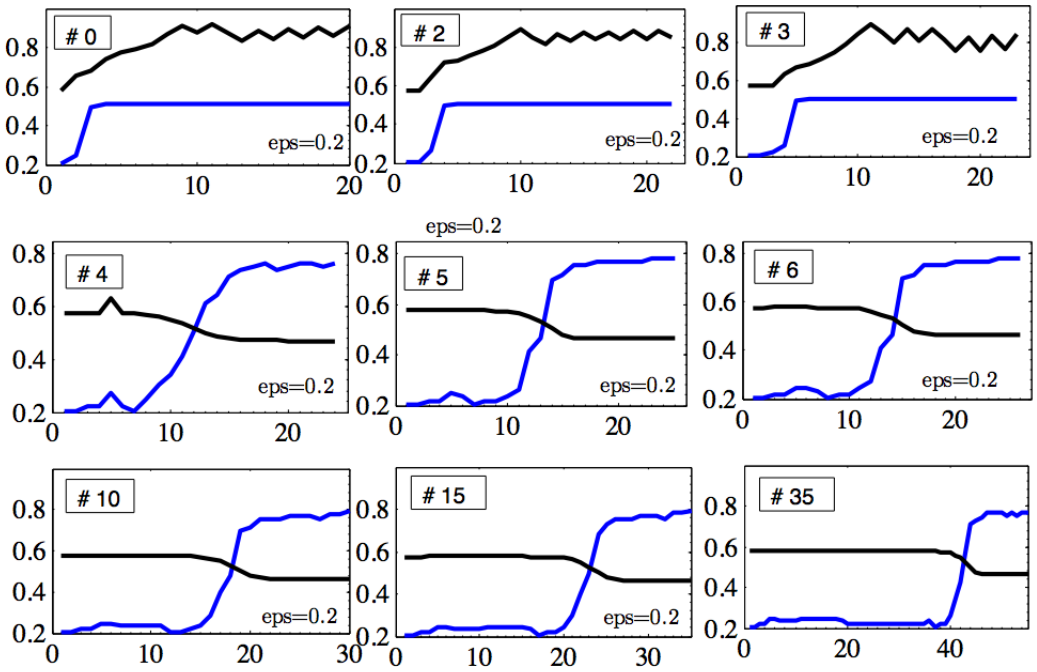

Figure 5.15: LIAS performance initiated with different size of the warm-up session. Learning curve is in black, and the blue curve is the accuracy rate.

to outliers and cutting off the computational effort spent on "poor" inputs, we propose a "pruning" procedure to omit the data points that happen to

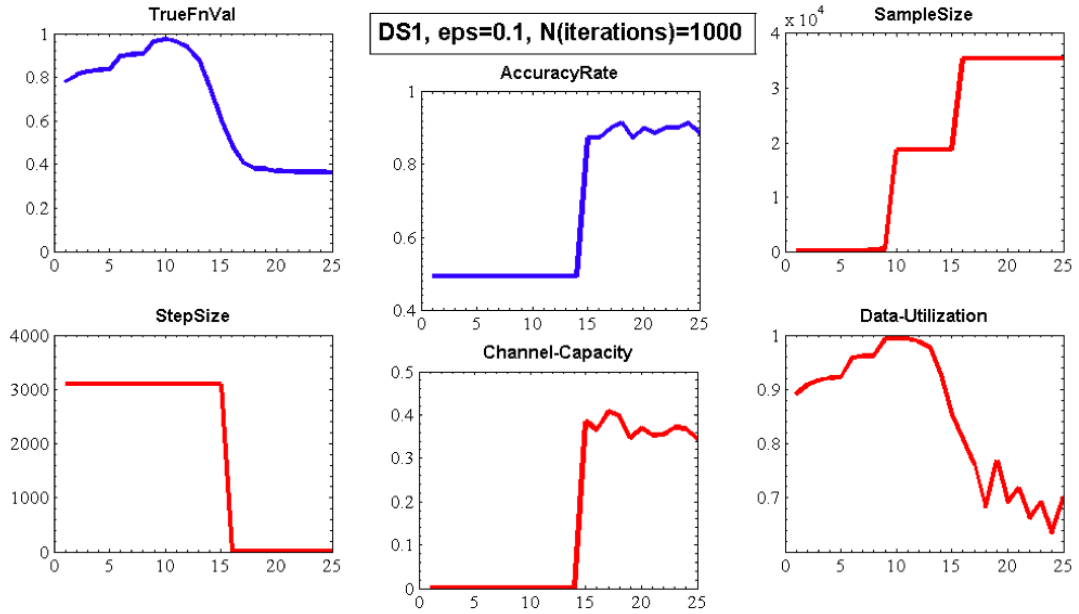

Figure 5.16: LIAS performance on DS1, under trend condition, with "perturbed K-Means" initialization.

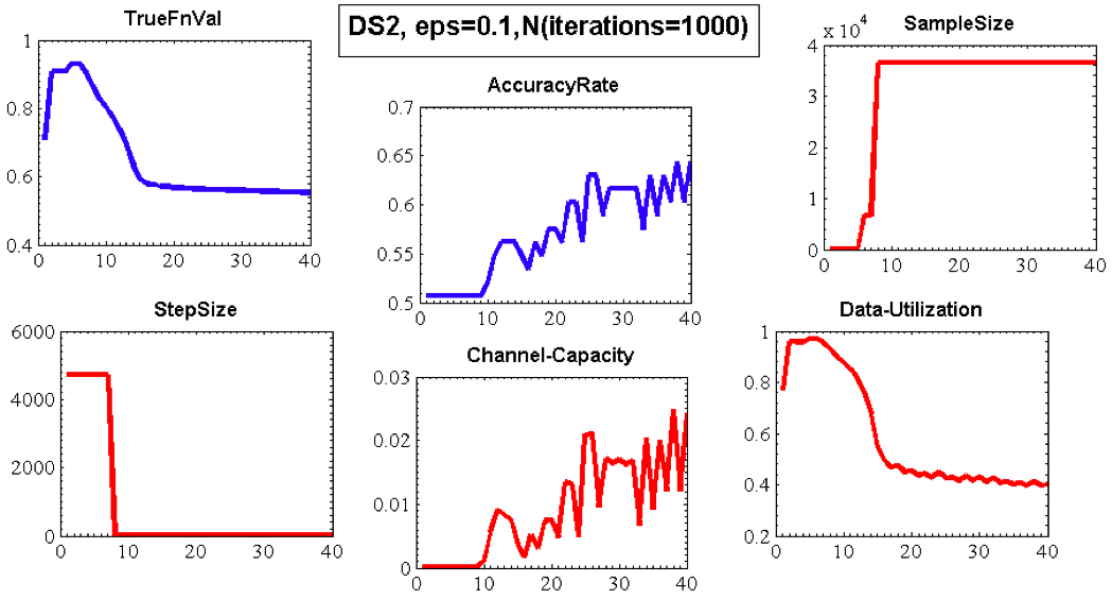

Figure 5.17: LIAS performance on DS2, under trend condition, with "perturbed K-Means" initialization.

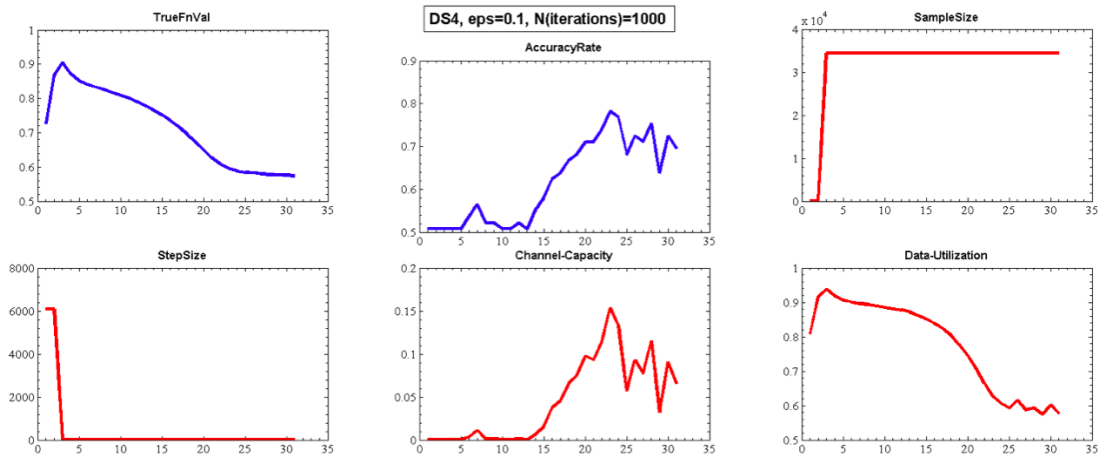

Figure 5.18: LIAS performance on DS4, under trend condition, with "perturbed K-Means" initialization.

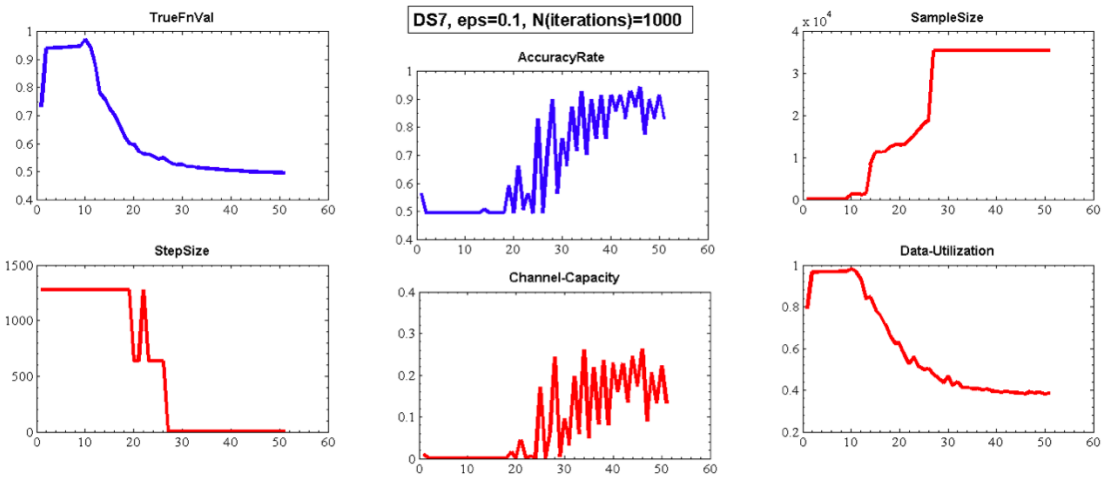

Figure 5.19: LIAS performance on DS7, under trend condition, with "perturbed K-Means" initialization.

possess highly different statistical properties. In this approach, a percentage of data with the minimum likelihood values calculated at the initial guess are eliminated from the data dictionary matrix introduced in Section 5.1. The rest of the data are then being used to refine the classifier within the training session. A similar care needs to be taken within the testing session: uncertainty in the output of classifiers should be quantified, and the set of test data with "poor" discriminative probabilities are subject to have the least effect on decision making. Accordingly a posterior probability threshold

T is utilized, to label the data as responsive if only the probability is above the specified threshold and the sequel output a prediction. This value is specifically chosen to be close to the average maximum posterior probabilities derived in the training session; although as demonstrated in Appendix 6.2, the performance is not quite sensitive to the choice of this parameter.

It is worthy to note that this condition causes delays within the prediction phases, when the posterior probabilities across motor actions are close. In the next section however, we show that such postpone meant happens only infrequently and that predictions by the proposed LIAS happens consistently; hence negligible interface idle time within BCI paradigm.

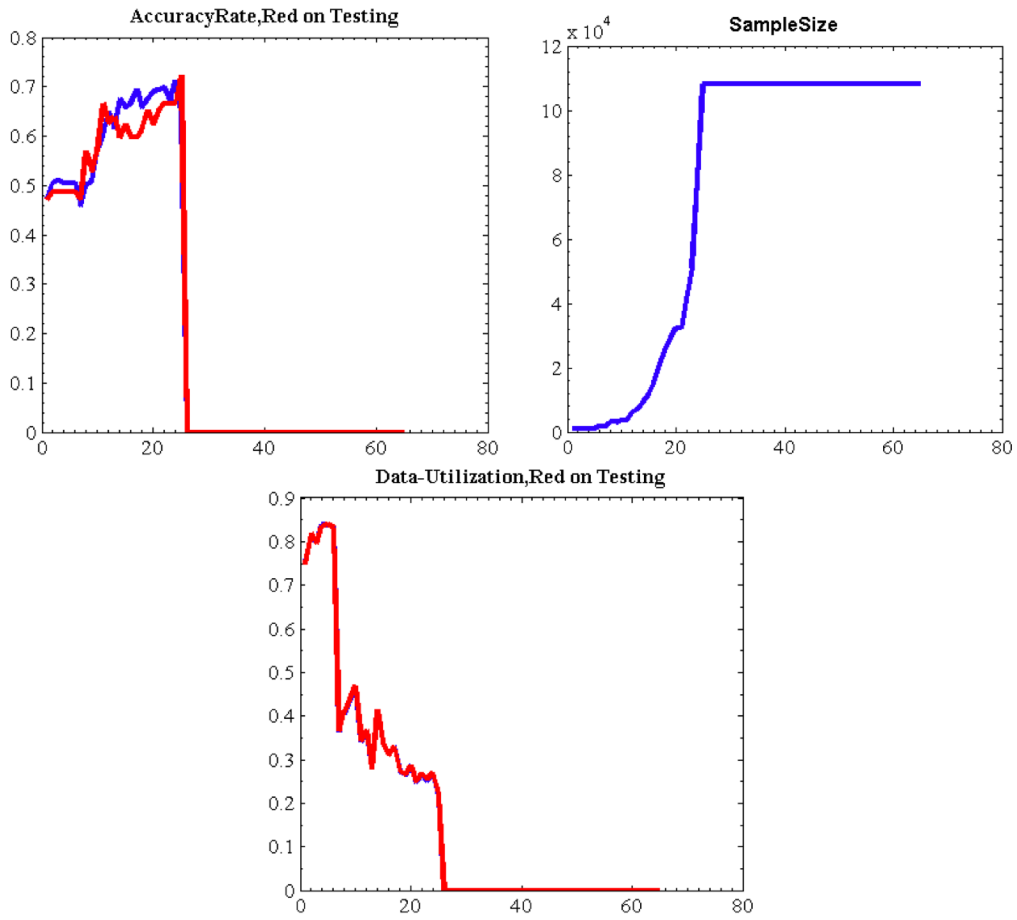

Figure 5.20: LIAS performance with ML model, implemented on DS6: after about 25 iterations, due to the coming outliers into the sample set, the overall loss is infinity and the data utilization shrinks to zero.

(H3) Choice of maximum step length for line search.

The theory of Adaptive-SCSR in Chapter 4 provides theoretical upper bound on the step length as in Theorem 4.4.1. Also we noted in Section 5.4.1, that due to lack of true curvature information, and hence unavailability of l_0 and l_1, we need to estimate the theoretical upper bound on the step length. This is done using the training data and estimating the true Hessian at the initial pattern. The maximum step length in line search is proposed as some scaler of the inverse maximum eigen value of the estimated Hessian. Please refer to Section 5.3 for the algorithm listing.

5.4.3 Performance Evaluation and Comparison

Activation of motor imagery, results in dynamic behavior in spatio-temporal EEG patterns. Our goal is to reach a Motor imagery BCI that can react to the changes in brain states over time, and classify the brain oscillatory dynamics in an online fashion. This would simplify be fitting an interface that is continuously responsive to the brain states.

Supervised classification methods are used to learn to recognize the patterns of EEG activities, and to classify features gathered in a dictionary matrix. The proposed BCI protocol has to perform two tasks: parameter estimation and classification. In an attempt to describe the properties of EEG patterns by a stylized HMM, LIAS is implemented to provide the parameter estimation routine, that follows with online classification of signals coming on the fly. Therefore the training phase outputs a design model for evaluation within the testing session. Given the testing EEG signals, the proposed algorithm paradigm outputs continuously a posterior prediction probability distribution on the set of motor actions. The motor action receiving the highest posterior probability is predicted to be the intended motor action of the subject. The output of the classification for a specific trial, is reported according to the majority vote, and the prediction accuracy is obtained as the fraction of the total number of trials in the "testing dataset" where the proposed algorithm made the correct prediction. In other words, the error rate relates to the number of mis-classified trials divided by the total number of trials. Fig 5.1 depicted the whole paradigm, and was fully discussed in Section 5.3.

Two other prominent competing methods are studied in this section, as Millan2004* and SunLuChen2011*. Millan2004* is slight variation of the method presented in [69] for EEG signal classification in an online control. Th parameter estimation method in this paper is that of stochastic gradient decent. SunLuChen2011* is also mimicking the method proposed in [104], whose parameter estimation approach is

BSG. We note that Adaptive-SCSR is performing between the two extreme lines in terms of sampling; while its performance is faster and the accuracy is significantly higher than the other two, in terms of computational effort. Accordingly, we study and compare the behavior of Adaptive-SCSR with SunLuChen2011*. Millan2004*, in terms of (i) speed of the interface, (ii) accuracy, and (iii) computational effort.

Speed of the BCI-interface

One of the most important criteria of a BCI, is the speed of the interface that is controlled by the classifier. A slow interface can cause user frustration and if considering prosthetic limb as a particular example of the interface, it cannot function as an actual limb.

Data utilization and channel capacity of the classifier, are the two metrics that can specify the speed of the interface. The more the testing data utilization, the faster the BCI can get. In other words, the interface remains idle while the data is being rejected and labeled as non-responsive. Hence rejecting so many testing data may degrade the performance of the interface, as it increases the idle time. However the probability of the unknown responses are going higher as getting closer to the local minima. More rigorously, given a fixed threshold on the posterior probabilities, testing data utilization is having a decreasing tendency when the accuracy is getting higher. Although this phenomenon initially seems counter intuitive, we observe that from those data that are reported as responsive (and included as part of data utilization), only a few of them would be labeled incorrectly with an improved quality training model. That is, often the classifier would either discriminate between the two classes correctly, or stays undecided. The pros is to make less wrong decisions. Invalid parameters (wrong hypothesis) would either result in unbounded loss, or wrong decisions. So with a poor parameter, higher number of testing data would contribute into decision making, but they lead the model to make wrong decisions. Therefore, there are two levels of decision making; one is to report the testing data as responsive or not-responsive, and second is to make accurate predictions. We may assign different scores to different decision making scenarios: -1 to when the data is reported as responsive, but yields wrong decision; 0 to when the data is not responsive; and 1 to when the data is responsive and yields correct prediction. Decision making with poor parameter usually takes the values of -1 and zero, but with a good parameter, it is taking the values of zero or one. So the total score of decision making with good parameter outweigh that of poor parameter. So with a good parameter, increasing the probability threshold for rejecting the data yields in more accurate predictions. That is, many of the lower accuracy predictions that pass the threshold could actually

be wrong. So we omit many of -1 scores, and put zeros instead. This can be effective specially in the case where the model is making fewer high-probability predictions (reported responsive data), but getting more of them correct than when the probability is lower. A useful metric to evaluate the interface's performance is the channel capacity, that considers the fraction of *correct* information that may be transmitted to the interface per unit time. We have observed that when updating the parameters, the channel capacity is mostly having the same trend as the accuracy rate (Fig. 5.21).

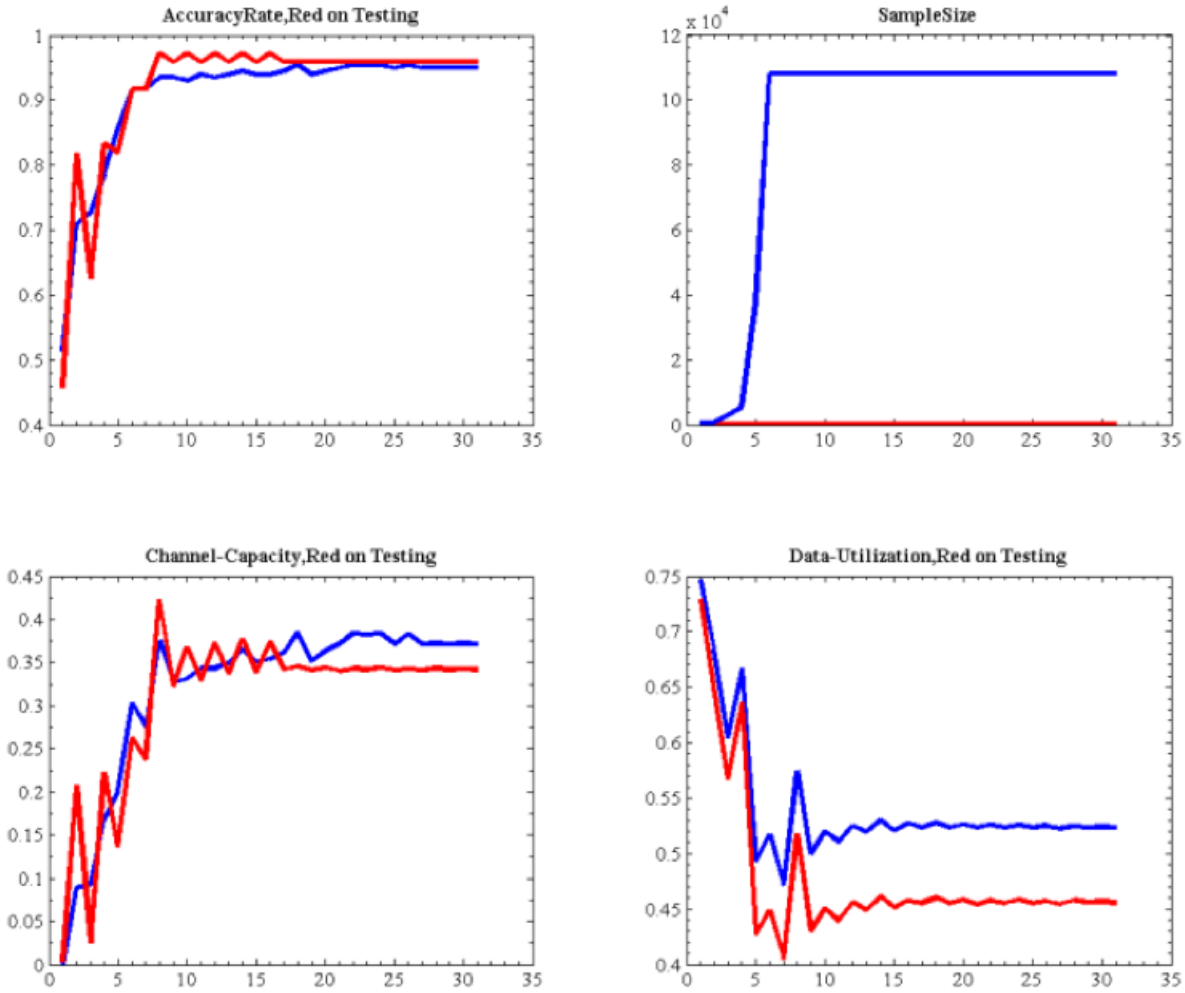

Figure 5.21: Typical performance of LIAS on different data sets.

Taking this trade-off into account, we have observed that the interface idle time would not exceed 0.1 of a second and the resulted interface is actually faster than that of competing methods (Table 5.2).

Table 5.2: Data Utilization on Different Probability Threshold

Threshold	K-Means	SunLuChen2011*	LIAS
0.65	0.31	0.22	0.42
0.7	0.2	0.13	0.27
0.75	0.12	0.08	0.15
0.8	0.07	0.05	0.08
0.85	0.04	0.03	0.04

Accuracy and Computational Effort

The other crucial factor considered within BCI paradigms, is the performance accuracy of the interface. Highly accurate performance, often avoids costly erroneous action of the interface, and hence improves user satisfaction.

Fig. 5.22 depicts a typical pattern across several datasets. LIAS reaches the local optimum with accuracy % 80, much faster than BSG. For a given total computational budget, LIAS routinely achieves prediction rates that are 20 - 30% higher than BSG and K-Means clustering (Figure 5.23).

Also reported in Table 5.3, are the prediction accuracy of the proposed paradigm alongside SunLuChen2011* and Millan2004*. As was noted earlier, the primary difference between SunLuChen2011*, Millan2004*, and the proposed paradigm lies in the choice of sample sizes across iterations when solving the parameter estimation problem. Recall that Millan2004* uses a regular SA method with the smallest possible sample size, while SunLuChen2011* uses a regular SA method with the highest possible sample size, that is the size of the training data provided. However, as discussed in Chapter 4, Adaptive-SCSR sequentially samples only as much data for function and derivative estimation as warranted by the inherent noise in the data and the estimated optimality of the current solution in the optimization routine. In other words, Adaptive-SCSR absorbs information from the dataset piecemeal, by adapting to the algorithm trajectory. This is in contrast to its primary competitor, BSG, which uses the entire dataset for every step in the optimization routine, and other state-of-the-art methods for predictive classification, e.g. Support Vector Machines (SVM), which either use a scaled- down dataset or have high memory complexity.

The strategic parsimony of Adaptive-SCSR guarantees asymptotic data-efficiency in a certain strict sense, without sacrificing predictive accuracy. Accordingly as can be seen in Table 5.3, the prediction accuracy of LIAS falls in the range 65 percent to 96 percent, with mean prediction accuracy across datasets estimated

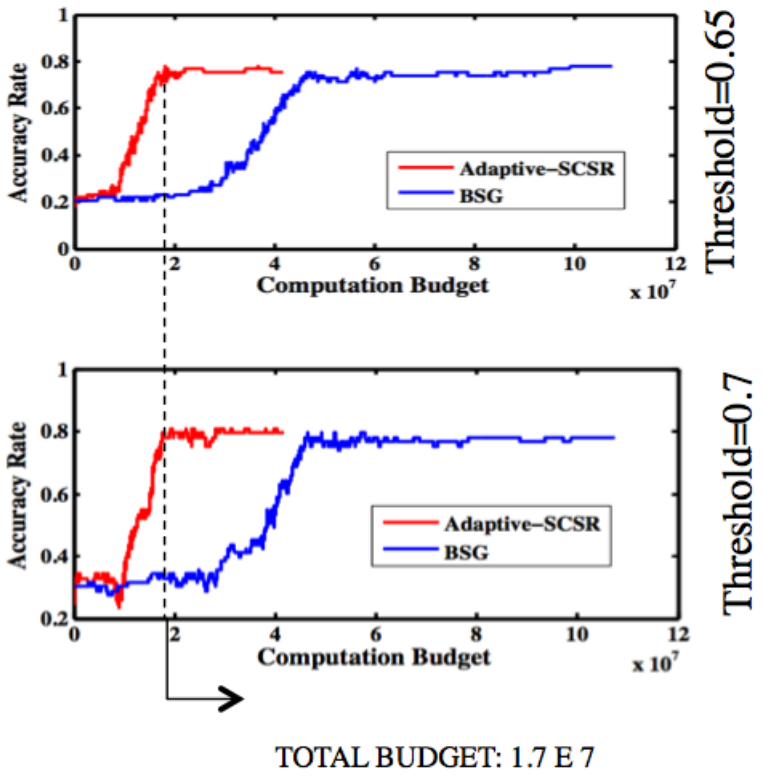

Figure 5.22: Adaptive-SCSR is shown to be computationally effective, relative to Batch Stochastic Gradient (BSG) methods.

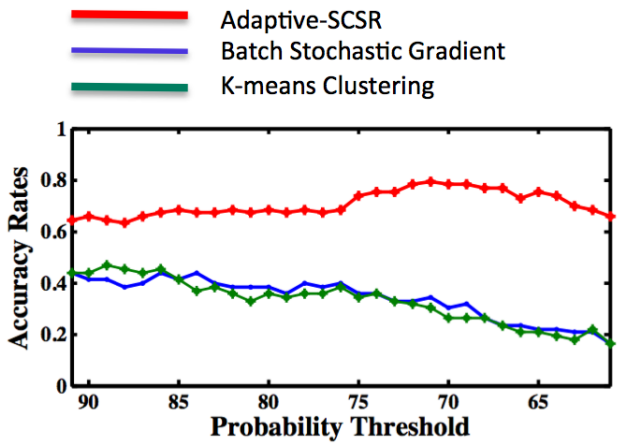

Figure 5.23: Accuracy rates for the cut-off budget = 1.7e7. LIAS gains 20% - 30% higher accuracy rates than the state-of-the-art methods.

to be approximately 76 percent. This is appreciably higher than Millan2004* that uses the smallest possible sample size, which clearly seems inadequate in terms of nullifying the natural variability in the EEG signals. What is surprising is the performance of SunLuChen2011*. In spite of using the highest possible sample size, the prediction rates of SunLuChen2011* seem lower than that of LIAS. This may be because of the well-established non-stationarity of EEG signals. (None of the three algorithms, SunLuChen2011*, Millan2004*, or Adaptive-SCSR, account for such non-stationarity explicitly.) The use of adaptive sampling in LIAS seems to (inadvertently) help with this issue.

Table 5.3: Percentage of Correctly Predicted Trials for Competing Methods

Data Set	K-Means	Millan2004*	SunLuChen2011*	LIAS
1	58.9	50.68	58.90	65.75
2	50.68	50.70	50.68	63.01
3	83.56	49.33	80.82	93.15
4	28.00	50.67	44.00	68.00
5	48.65	49.99	50.70	63.51
6	39.47	51.31	47.37	64.47
7	35.62	50.68	32.88	82.19
8	57.53	49.32	89.04	95.89
9	72.22	50.00	86.11	88.89
Mean	52.73	50.29	60.27	76.1

Furthermore, LIAS augmented with ML criterion, is compared with ensemble SVM classifier, proposed by the winner in BCI competition 2008 [65], over the experiments with 7 human subjects. We observed that in most of the data sets, LIAS obtains higher accuracy rates, while using only half of the data utilized in ensemble SVM.

5.5 Concluding Remarks

The fast and accurate interpretation of EEG signals from the human brain brings forth tremendous application opportunities. One such opportunity is the creation of customized and thought-controlled prosthetic devices that can be "worn" by severely motor impaired patients. While this idea has existed in a conceptual form for about a decade, actual implementation has been stymied by the need to process large amounts of EEG data in an online context towards predicting patient intent.

Table 5.4: Comparison of LIAS with Ensemble SVM : While being twice computationally effective, LIAS is observed to often outperform ensemble SVM proposed by the winner of 2008 BCI competition.

Data Set	Ensemble SVM	LIAS
1	87.5	82.1
2	56.8	60.3
4	63.6	68.5
5	58	58
6	77.1	85
8	94.3	96
9	93.9	90.2

In this chapter, we have laid the groundwork for a fundamentally new algorithm that seems to thwart a major roadblock to online implementation of EEG classifiers. Specifically, by combining an online learning model with a stochastic recursion that parsimoniously but adequately samples EEG data, we are able to devise an algorithm that yields good prediction rates at near instantaneous speeds, modulo paying attention to certain implementation details. Our extensive numerical experience seems to suggest that the proposed algorithm performs comparably with the best available classifiers (as measured by prediction rates) at times that are one to two orders of magnitude lower.

The obvious next step is understanding whether the proposed algorithm scales well to contexts where the prosthetic in use has the ability to perform a wide array of tasks. Apart from this, we are exploring newer paradigms where a patient continuously "wears and trains" with a prosthetic device, much like the way a child learns to perform a motor task.

Chapter 6

Final Remarks

6.1 Concluding Remarks

The use of simulation-based estimators within well-established algorithmic recursions is becoming an attractive paradigm to solve optimization and root-finding problems in contexts where the underlying functions can only be estimated. In such contexts, the question of how much to simulate (towards estimating function and derivative values at a point) becomes important particularly when the available simulations are computationally expensive.

In this dissertation, we have laid the groundwork for a fundamentally new sampling based method that seems to thwart a major roadblock to "online" parameter estimation within simulation optimization problems. Towards exposition of the proposed methodology, we first introduced Sampling Controlled Stochastic Recursions (SCSR) in Chapter 3, and characterized an interplay between the structural error inherent in the recursion in use and the sampling error inherent in the simulation estimator. This provided guidance (see Figure 3.1) on the *rate* at which sampling should be performed in order that SCSR's iterates remain efficient. While still ensuring the increase rates prescribed by our results in Chapter 3, Chapter 4 proposed an implementable algorithm that dynamically chooses sample sizes as a function of the observed trajectory of the algorithm. This points to our main contribution and highlights the main distinction between the proposed methodology in this dissertation, and the state-of-the art algorithms for solving SO problems.

Within traditional algorithms like SAA and SA, typically the sample size is either fixed or its growth rate follows a deterministic rule. For instance [13, 11, 12] investigate sampling based methods in stochastic programs under SAA, when the

sampling schedules follow a linear/sub-linear rate function. In [13], a near-optimal solution is of interest under fixed width stopping rules; whereas [11, 12] use relative width interval estimators, and the solution to SAA is reported when the point estimator of the "optimality gap" falls below a fraction of sampling variability. In addition, in the context of sampling controlled SA, variants of "two stage sequential" stopping rules [25] and "fully sequential" stopping rules [85] are implemented . Within two-stage sequential stopping rules, the sample size for each iteration is determined based on the history of the method, in an "off-line" fashion. Therefore using such sampling strategy within stochastic recursion, the iterative sample size is being "calculated" and used in the current iteration. However in fully-sequential stopping rules, the samples are observed "on the fly" and the stopping rule decides on the sample size based on the statistical properties of coming data. Therefore the resulted sample size in fully-sequential methods is a random variable, whereas in two-stage procedures the sample size is static and fixed. In [25], a two-stage sequential stopping rule is derived as a result of generating descent directions *with high probability*. The idea is motivated by showing that an iterative guarantee for descent direction condition, results in almost sure convergence, with complexity estimate as good as that of the stochastic gradient descent method. However the proposed descent direction condition cannot be checked explicitly, as the true value for the gradient cannot be computed. Therefore to obtain an "implementable" method, it is replaced with an alternative two-stage sampling rule that ensures the same condition "sufficiently often". In [85] on the other hand, a fully-sequential stopping rule is devised to increase the sample size until making sure that the deviation of the estimator from the true solution is more due to the bias of the estimator with respect to the true solution than the sampling variability. We note that both algorithms proposed in [25] and [85] are sampling heuristics, lacking theoretical support on asymptotic properties of the resulted iterates.

Adaptive-SCSR proposed in Chapter 4 formulated an easy and implementable shift from deterministic to stochastic sample size increase, which open the door for algorithms that achieve finite-time efficiency through (stochastic) sampling, while retaining asymptotic efficiency. The fully sequential stopping rule imposed sampling until the standard error estimate (of the object being estimated within the recursion) is in lock step with the estimate itself. Due to fully sequential properties, the sampling rule decides either to stop or continue sampling based on online observations, which suggests a good fit to statistical properties of data. Under this strategy, consistency imposes a very loose lower bound on the sampling rate, while efficiency imposes maximum and minimum rates that depend on the quality of the deterministic recursion. The minimal rate function for sampling is called "escorting sequence" (denoted by ν in Chapter 4), as it drags the iterates to the vicinity of the root. Another important component in the rule is defined as

"coercion constant" (denoted by ε in Chapter 4) to make sure that the sampling error drops at the requisite rate. We have showed however, that the behavior of the iterates are not too sensitive to the choice of these parameters, and optimality of SCSR augmented with the proposed sequential rule follows under weak conditions on these two parameters. Interestingly this makes the discrepancy between theory and the true algorithm used in practice, relatively small.

Eventually in Chapter 5 Adaptive-SCSR augmented with a line search procedure is implemented to solve a non-trivial parameter estimation problem for an online learning model (section 5.2). The algorithm paradigm is called Line-search Adaptive-SCSR (LIAS) and is implemented within the context of online classification of human brain signals towards constructing a Brain-Computer Interface (BCI). The sequel parsimoniously but adequately samples EEG data (described in section 5.1), and the classifier yields good prediction rates at near instantaneous speeds, modulo paying attention to certain implementation details. To the best of our knowledge, the crux of dynamic EEG sampling in the context of online EEG pattern prediction has not been explored within the stream of machine-learning literature on BCI. So our work points to a proof of concept here. Also our extensive numerical experience seems to suggest that the proposed algorithm performs comparably with the best available classifiers (as measured by prediction rates) at times that are one to two orders of magnitude lower. This fast and accurate interpretation of EEG signals from the human brain may bring forth tremendous application opportunities. One such opportunity is the development of a variety of BCI applications, e.g. thought-controlled prosthetic devices that can be worn by severely motor impaired patients. The benefits of using such an "optimized" estimation scheme within BCI applications are four-fold: i) for the same limited amount of EEG data in our data set, our procedure achieves up to 20-30% higher correct prediction rate than the current state-of-the-art methods, ii) in the context of online learning as data is being collected, the adaptive use of EEG trials reduces the length of the training phase that the subject needs to undergo, thus also reducing the impact of secondary factors like fatigue and loss of attention iii) the parsimony of our method reduces the load on critical EEG testing resources and personnel per individual trained, and iv) the finite-time speed and accuracy of the method makes it a fit choice for advanced use such as adaptive re-training of the classification model.

6.2 Future Research

Four main set of improvements are of interest: (1) generalization of the optimization problem 1.1 under consideration; (2) generalization of Adaptive-SCSR sampling rule to a "family" of optimal sequential sampling rules; (3) advancements over EEG pattern recognition for BCI applications; (4) development of EEG pattern recognition for disease diagnosis applications.

(1) **Generalization of (1.1).**

While Adaptive-SCSR is devised as a parameter-free and easy-to-implement methodology for solving SO problems, there are still limitations due to the structural conditions on the true objective function in problem (1.1). Of our current research questions, is the possibility to drop strong convexity assumption (4.2.2) on the objective function. We anticipate that local Lipschitz conditions, would be of good alternatives to dilute strong convexity assumption, and still retain the same theoretical results. This is particularly of interest when we observe in practice that due to problem-specific modeling aspects, the underlying objective function may not even possess convexity. However, in case of multi-modality of the objective function, we may assume "piecewise convexity", and take a random-restart procedure that seek for multiple local optimums. This approach is successfully attempted in Chapter 5, where the multi-modal objective function is constructed based upon a Gaussian mixture model. Of other interesting simplifications, we consider taking dependent and even "non-stationary" data, instead of iid observations assumed in problem (1.1). When a common stream of random numbers is used in calculating the gradient estimates, the input data to the optimization problem becomes dependent. In addition, in a variety of applications with online parameter estimation, the sample path may exhibit non-stationary features like Heteroscedasticity. In such cases, the statistical properties of data are changing in time, hence the true solution to problem (1.1) is not unique. In other words, as properties of data vary over time, the problem (1.1) is changing accordingly. A weaker situation is when there are only finite number of scenarios, and the optimization problem is required to "pick" the true solution as data is observed during the time. This particular phenomenon is of future research interests, specially in the context of online EEG classification, where the nonstationary EEG data can be modeled using pairwise Markov models and triplet Markov models.

(2) **A family of optimal sequential sampling rules.**

According to the results presented in Chapter 3 and 4, three main components determine the quality of the SCSR iterates: the speed of the recursion, the quality of the sampled estimator, and the minimal rate of sampling. According to the results in Chapter 3, the speed of recursion determines optimal rate of sampling, with faster recursions allowing for a wider range of sample sizes while remaining efficient. In other words, given a specific recursion, any sampling regime whose asymptotic growth rate yields canonical rate of convergence for SCSR (Fig. 3.1), is an ideal sampling approach to be attached to the stochastic recursion. However as observed through the main results in Chapter 4, this asymptotic behavior of sample size, is intimately linked with the minimal rate of sampling and the quality of the sampled estimator in SCSR. We showed that under weak conditions on the minimal rate of sampling, strong consistency of SCSR is ensured, which serves as a baseline for analysis of the asymptotic sampling rate. The main criterion to gain the quality of the sampled estimator is demonstrated through the following two main set of results:

(i) Theorems 4.4.2 and 4.4.3, that showed the stochastic sample size "concentrates" around $h^{-\eta}$ in probability and expectation, where h was the true gradient and $\eta = \frac{2}{1-2\varepsilon}$ for $0 < \varepsilon < 1/2$.

(ii) Theorem 4.4.4, which proved that for our sampling scheme that is claimed to be "optimal", the sampling error ($\mathbb{E}[\|\tilde{H}(M, X) - h(X)\|^2]$) and the recursion error $\|h(X)\|^2$ are balanced asymptotically. That is, when $\|h(X)\|$ is arbitrarily close to zero, $\frac{\mathbb{E}[\|\tilde{H}(M,X)-h(X)\|^2]}{\|h(X)\|^2} \approx 1$.

Interestingly the similar sort of results to (i) and (ii) may be perceived in sequential statistics when risk-efficiency of the sampling stopping rule is of interest. This connection can be spelled out as follows.

Recall the main problem statement in risk-efficient sequential statistics described in section 4.3. Given $\bar{X}_m$ as the sample mean over m iid observations from a population, and μ as the true mean of the population, the typical loss structure is $L_m := a(\bar{X}_m - \mu)^2 + \lambda m$, where a is some scaler of the true variance of the population, and λ is a constant that is allowed to approach zero. The "risk" is defined as $R_m := \mathbb{E}L_m$. First assuming the variance is known as σ^2, we have $R_m = \frac{a\sigma^2}{m} + \lambda m$. Letting m_0 be the sample size that minimizes the risk, we get $R_{m_0} = \Psi(\sqrt{\lambda})$. Now assuming unknown variance, we adopt a sequential sampling rule to determine the sample size. Letting the sample size be M when the rule stops, a risk-efficient sequential rule is to ensure $\lim_{\lambda\to 0} \frac{R_M}{R_{m_0}} = \frac{a\mathbb{E}[(\bar{X}_M-\mu)^2]+\lambda M}{\Psi(\sqrt{\lambda})} = 1$. Loosely speaking, as $\lambda \to 0$, we need to have (i') $M \approx \lambda^{-1/2}$, and (ii') $\mathbb{E}[(\bar{X}_M - \mu)^2] \approx \sqrt{\lambda}$. Now

the connection of (i') & (ii') to (i) & (ii) may be perceived when considering $\bar{X}_M := H(M, X)$, $\sqrt{\lambda} := \|h(X)\|^2$, and the fact that M "concentrates" around $h^{-\eta}$, for η arbitrarily close to 2.

Highlighting this connection, we envision that given any type of recursion in hand, there exist a "family" of risk-efficient sequential sampling rules that retain efficiency of SCSR by ensuring specific properties for the sampled estimator similar to (i) and (ii). This family would exclude non-optimal sampling regimes when the ε-value within Definition 1.4.1 is prevented to be arbitrarily close to zero in order to ensure aforementioned conditions. Theorem 4.4.7) provided an example of such "non-optimal" sampling regime when Adaptive-SCSR is used as an instance of a family of sampling regimes.

(3) **Advancements over EEG pattern recognition for BCI applications.**

Moreover, within BCI applications, understanding whether the proposed algorithm scales well to multi-task BCIs is of future research interests. This points to the application where the interface has the ability to perform a wide array of tasks, rather than only two at a time. Apart from this, we are exploring newer paradigms where a subject continuously test and trains with the interface, much like the way a child learns to perform a motor task.

(4) **Development of EEG pattern recognition for disease diagnosis applications.**

Beside the application of EEG pattern recognition in BCI, we are also interested in disease diagnosis applications. For instance one of the main applications of EEG pattern recognition is to detect status epilepticus among ICU patients. Speed and accuracy of the detection are highly crucial in such applications. It is observed that the longer the duration of status epilepticus, the higher the risk of death and disability among ICU patients, independent of multiple potential confounding variables. Likewise, when false negative detection, Seizure cannot be detected in spite of monitoring, hence missing early treatment opportunity, and increasing risk of death and disability. The main bottle neck in disease diagnosis applications however, is that the training EEG trials are often unlabeled, and hence required to be treated with unsupervised learning through hidden Markov models. Therefore it is appealing to us to evaluate performance of our classifier in terms of speed and accuracy, in the context of unsupervised learning, and accordingly proceed to apply the proposed learning scheme for disease diagnosis.

Bibliography

Bibliography

[1] O. Alagoz, A. J. Schaefer, and M. S. Roberts. Optimization in organ allocation. In P. Pardalos and E. Romeijn, editors, *Handbook of Optimization in Medicine*. Kluwer Academic Publishers, 2009.

[2] A. Alkan and S. B. Akben. Use of k-means clustering in migraine detection by using eeg records under flash stimulation. *International Journal of the Physical Science*, 6(4):641–650, 2011.

[3] G. Alsmeyer. On the moments of certain first passage times for linear growth processes. *Stochastic processes and their applications*, 25:109–136, 1987.

[4] O. AlZoubi, I. Koprinska, and R. A. Calvo. Classification of brain-computer interface data. In *Proceedings of the 7th Australasian Data Mining Conference-Volume 87*, pages 123–131. Australian Computer Society, Inc., 2008.

[5] F. J. Anscombe and J. Francis. Sequential estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–29, 1953.

[6] S. Asmussen and P. W. Glynn. *Stochastic simulation: Algorithms and analysis*, volume 57. Springer Science & Business Media, 2007.

[7] S. Asmussen and P. W. Glynn. *Stochastic Simulation: Algorithms and Analysis*. Springer, New York, NY., 2007.

[8] J. Atlason, M. A. Epelman, and S. G. Henderson. Optimizing call center staffing using simulation and analytic center cutting plane methods. *Management Science*, 54(2):295–309, 2008.

[9] R. Babuka, P. Van der Veen, and U. Kaymak. Improved covariance estimation for Gustafson-Kessel clustering. In *Fuzzy Systems, 2002. FUZZ-IEEE'02. Proceedings of the 2002 IEEE International Conference on*, volume 2, pages 1081–1085. IEEE, 2002.

[10] D. Basu. On statistics independent of a complete sufficient statistic. *Sankhyā: The Indian Journal of Statistics*, pages 377–380, 1955.

[11] G. Bayraksan and D. P. Morton. Assessing solution quality in stochastic programs. *Mathematical Programming Series B*, 108:495–514, 2007.

[12] G. Bayraksan and D. P. Morton. A sequential sampling procedure for stochastic programming. *Operations Research*, 59(4):898–913, 2009.

[13] G. Bayraksan and P. Pierre-Louis. Fixed-width sequential stopping rules for a class of stochastic programs. *SIAM Journal on Optimization*, 22(4):1518–1548, 2012.

[14] M. S. Bazaara, H. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, New York, NY., 2006.

[15] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear programming: theory and algorithms*. John Wiley & Sons, 2013.

[16] P. Billingsley. *Probability and Measure*. Wiley, New York, NY., 1995.

[17] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kübler, J. Perelmouter, E. Taub, and H. Flor. A spelling device for the paralysed. *Nature*, 398(6725):297–298, 1999.

[18] B. Blankertz, G. Curio, and K. R. Muller. Classifying single trial eeg: Towards brain computer interfacing. *Advances in neural information processing systems*, 1:157–164, 2002.

[19] J. Blum. Multidimensional stochastic approximation. *Annals of Mathematical Statistics*, 25(4):737–744, 1954.

[20] V. S. Borkar. Stochastic approcimation with two time scale. *systema dn Control Letters*, 29:291–294, 1997.

[21] V. S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Cambridge University Press, Cambridge, UK, 2008.

[22] M. Broadie, D. M. Cicek, and A. Zeevi. An adaptive multidimensional version of the Kiefer-Wolfowitz stochastic approximation algorithm. In M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, editors, *Proceedings of the 2009 Winter Simulation Conference*, pages 601–612. Institute of Electrical and Electronics Engineers: Piscataway, New Jersey, 2009.

[23] M. Broadie, D. M. Cicek, and A. Zeevi. General bounds and finite-time improvement for the kiefer-wolfowitz stochastic approximation algorithm. *Operations Research*, 59:1211–1224, 2010. To appear.

[24] C. Brunner, R. Leeb, G. Müller-Putz, A. Schlögl, and G. Pfurtscheller. Bci competition 2008–graz data set a. *Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology*, 2008.

[25] R. H. Byrd, G. M. Chin, J. Nocedal, and Y. Wu. Sample size selection in optimization methods for machine learning. *Mathematical programming*, 134(1):127–155, 2012.

[26] J. K. Chapin, K. A. Moxon, R. S. Markowitz, and M. A. Nicolelis. Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex. *Nature neuroscience*, 2(7):664–670, 1999.

[27] Y. S. Chow and H. Robbins. On the asymptotic theory of fixed-width sequential confidence intervals for the mean. *The Annals of Mathematical Statistics*, pages 457–462, 1965.

[28] Y. S. Chow and K. F. Yu. The performance of a sequential procedure for the estimation of the mean. *The Annals of Statistics*, pages 184–189, 1981.

[29] M. Chu, Y. Zinchenko, S. G. Henderson, and M. B. Sharpe. Robust optimization for intensity modulated radiation therapy treatment planning under uncertainty. *Physics in Medicine and Biology*, 50:5463–5477, 2005.

[30] K. L. Chung. On a stochastic approximation method. *Annals of Mathematical Statistics*, 25:463–483, 1954.

[31] A. Dembo and O. Zeitouni. *Large Deviations Techniques and Applications*. Springer-Verlag, New York, NY., 1998.

[32] G. Deng and M. C. Ferris. Adaptation of the UOBQYA algorithm for noisy functions. In L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, editors, *Proceedings of the 2006 Winter Simulation Conference*, pages 312–319. Institute of Electrical and Electronics Engineers: Piscataway, New Jersey, 2006.

[33] G. Deng and M. C. Ferris. Variable-number sample-path optimization. *Mathematical Programming*, (117):81–109, 2009.

[34] C. Derman. An application of Chung's lemma to the Kiefer-Wolfowitz stochastic approximation procedure. *Annals of Mathematical Statistics*, 27:532–536, 1956.

[35] J. Dippon and J. Renz. Weighted means in stochastic approximation of minima. *SIAM Journal on Control and Optimization*, 35:1811–1827, 1997.

[36] R. Douc, G. Fort, E. Moulines, and P. Soulier. Practical drift conditions for subgeometric rates of convergence. *Annals of Applied Probability*, pages 1353–1377, 2004.

[37] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 999999:2121–2159, 2011.

[38] P. Dupuis and R. Simha. On sampling controlled stochastic approximation. *IEEE Transactions on Automatic Control*, 36(8):915–924, 1991.

[39] T. Ebrahimi, J. M. Vesin, and G. Garcia. Brain-computer interface in multimedia communication. *Signal Processing Magazine, IEEE*, 20(1):14–24, 2003.

[40] V. Fabian. On asymptotic normality in stochastic approximation. *Annals of Mathematical Statistics*, 39:1327–1332, 1968.

[41] M. Figueiredo and A. K. Jain. Unsupervised learning of finite mixture models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(3):381–396, 2002.

[42] I. G. Costa Filho. *Mixture Models for the Analysis of Gene Expression: Integration of Multiple Experiments and Cluster Validation*. PhD thesis, Freie Universität Berlin, 2008.

[43] E. W. Frees and D. Ruppert. Estimation following a Robbins-Monro designed experiment. *Journal of American Statistical Association*, 85:1123–1129, 1990.

[44] J. Galombos and E. Seneta. Regularly varying sequences. *Proceedings of the American Mathematical Society*, 41(4):110–116, 2008.

[45] M. Ghosh and N. Mukhopadhyay. Sequential point estimation of the mean when the distribution is unspecified. *Communications in Statistics-Theory and Methods*, 8(7):637–652, 1979.

[46] F. Gonzalez, D. Dasgupta, and R. Kozma. Combining negative selection and classification techniques for anomaly detection. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 1, pages 705–710. IEEE, 2002.

[47] S. G. Henderson and B. L. Nelson, editors. volume 13 of *Handbooks in Operations Research and Management Science: Simulation*. Elsevier, 2006.

[48] Y. T. Herer, , M. Tzur, and E. Yucesan. The multilocation transshipment problem. *IIE Transactions*, 38:185–200, 2006.

[49] L.R. Hochberg, M.D. Serruya, G. M. Friehs, J.A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, and J. P. Donoghue. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature*, 442(7099):164–171, 2006.

[50] T. Homem-de-Mello, A. Shapiro, and M. L. Spearman. Finding optimal release times using simulation based optimization. *Management Science*, 45:86–102, 1999.

[51] Y. Huang, K. B. Englehart, B. Hudgins, and A. Chan. A gaussian mixture model based classification scheme for myoelectric control of powered upper limb prostheses. *Biomedical Engineering, IEEE Transactions on*, 52(11):1801–1811, 2005.

[52] R. Huber, M. F. Ghilardi, M. Massimini, and G. Tononi. Local sleep and learning. *Nature*, 430(6995):78–81, 2004.

[53] P. R. Kennedy, R. A. Bakay, M. M. Moore, K. Adams, and J. Goldwaithe. Direct control of a computer from the human central nervous system. *Rehabilitation Engineering, IEEE Transactions on*, 8(2):198–202, 2000.

[54] H. Kesten. Accelerated stochastic approximation. *Annals of Mathematical Statistics*, 21:41–59, 1958.

[55] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics*, 23:462–466, 1952.

[56] S. Kim, R. Pasupathy, and S. G. Henderson. A guide to SAA. Frederick Hilliers OR Series. Elsevier, 2012.

[57] H. J. Kushner and G. G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer-Verlag, New York, NY., 2003.

[58] A. M. Law. *Simulation Modeling and Analysis*. McGraw-Hill, New York, NY., 2007.

[59] A. M. Law, W. D. Kelton, and L. W. Koenig. Relative width sequential confidence intervals for the mean: Relative width sequential confidence intervals. *Communications in Statistics-Simulation and Computation*, 10(1):29–39, 1981.

[60] K. Le and R. Pasupathy. A note on the number of random restarts required to approximate All solutions of a stochastic nonlinear system. *Operations Research*, 2011. To appear.

[61] M. A. Lebedev and M. A. Nicolelis. Brain–machine interfaces: past, present and future. *TRENDS in Neurosciences*, 29(9):536–546, 2006.

[62] S. Lemm, B. Blankertz, G. Curio, and K. R. Muller. Spatio-spectral filters for improving the classification of single trial eeg. *Biomedical Engineering, IEEE Transactions on*, 52(9):1541–1548, 2005.

[63] E. C. Leuthardt, G. Schalk, J. R. Wolpaw, J. G. Ojemann, and D. W. Moran. A brain–computer interface using electrocorticographic signals in humans. *Journal of neural engineering*, 1(2):63, 2004.

[64] P. Li, M. Abbas, and R. Pasupathy. Simulation-based optimization of maximum green setting under retrospective approximation framework. *Transportation Research*, 2010. To appear.

[65] S. R. Liyanage, C. Guan, H. Zhang, K. K. Ang, J. Xu, and T. H. Lee. Dynamically weighted ensemble classification for non-stationary eeg processing. *Journal of neural engineering*, 10(3):036007, 2013.

[66] M. D. McKay, R. J. Beckman, and W. J. Conover. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.

[67] J. R. Millán, F. Renkens, J. Mourino, and W. Gerstner. Non-invasive brain-actuated control of a mobile robot. In *IJCAI*, pages 1121–1126, 2003.

[68] J. R. Millán, F. Renkens, J. Mouriño, and W. Gerstner. Brain-actuated interaction. *Artificial Intelligence*, 159(1):241–259, 2004.

[69] J. R. Millan, F. Renkens, J. Mouriño, and W. Gerstner. Noninvasive brain-actuated control of a mobile robot by human eeg. *Biomedical Engineering, IEEE Transactions on*, 51(6):1026–1033, 2004.

[70] A. Mokkadem and M. Pelletier. A generalization of the averaging procedure: the use of two-time-scale algorithms. *SIAM Journal on Control and Optimization*, 49(4):1523–1543, 2011.

[71] A. Mokkadem and M. Pelletier. A generalization of the averaging procedure: The use of two-time-scale algorithms. *SIAM Journal on Control and Optimization*, 49:1523, 2011.

[72] N. Mukhopadhyay and S. Datta. On sequential fixed-width confidence intervals for the mean and second-order expansions of the associated coverage probabilities. *Annals of the Institute of Statistical Mathematics*, 48(3):497–507, 1996.

[73] K. R. Müller, M. Tangermann, G. Dornhege, M. Krauledat, G. Curio, and B. Blankertz. Machine learning for real-time single-trial eeg-analysis: from brain–computer interfacing to mental state monitoring. *Journal of neuroscience methods*, 167(1):82–90, 2008.

[74] S. Musallam, B. D. Corneil, B. Greger, H. Scherberger, and R. A. Andersen. Cognitive control signals for neural prosthetics. *Science*, 305(5681):258–262, 2004.

[75] 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.

[76] Y. Nesterov. *Introductory Lectures on Convex Optimization*. Kluwer Academic Publishers, Norwell, MA, 2004.

[77] M. A. Nicolelis. Brain–machine interfaces to restore motor function and probe neural circuits. *Nature Reviews Neuroscience*, 4(5):417–422, 2003.

[78] M. A. Nicolelis and J. K. Chapin. Controlling robots with the mind. *SCIENTIFIC AMERICAN-AMERICAN EDITION-*, 287(4):46–55, 2002.

[79] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[80] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, NY., 1970.

[81] R. Pasupathy. On choosing parameters in retrospective-approximation algorithms for stochastic root finding and simulation optimization. *Operations Research*, 58:889–901, 2010.

[82] R. Pasupathy and S. Ghosh. Simulation optimization: a concise overview and implementation guide. *TutORials in Operations Research*, pages 122–150, 2013.

[83] R. Pasupathy and S. G. Henderson. In *Proc. of the 2011 Winter Simulation Conference*, Piscataway, NJ.

[84] R. Pasupathy and S. Kim. The stochastic root-finding problem: overview, solutions, and open questions. *ACM TOMACS*, 21(3):19, 2011.

[85] R. Pasupathy and B. W. Schmeiser. DARTS — dynamic adaptive random target shooting. In B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, editors, *Proceedings of the 2010 Winter Simulation Conference*. Institute of Electrical and Electronics Engineers: Piscataway, New Jersey.

[86] R. Pasupathy and B. W. Schmeiser. Retrospective-approximation algorithms for multidimensional stochastic root-finding problems. *ACM TOMACS*, 19(2):5:1–5:36, 2009.

[87] G. Ch. Pflug. Stochastic optimization and statistical inference. In A. Ruszczynski and Shapiro, editors, *Stochastic Programming*, Handbooks in Operations Research and Management Science, pages 427–482. Elsevier, 2004.

[88] B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

[89] J. Rafiee, M.A. Rafiee, N. Prause, and M.P. Schoen. Wavelet basis functions in biomedical signal processing. *Expert Systems with Applications*, 38(5):6190–6201, 2011.

[90] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[91] D. Ruppert. A Newton-Raphson version of the multivariate Robbins-Monro procedure. *Annals of Statistics*, 13:236–245, 1985.

[92] D. Ruppert. Stochastic approximation. Handbook in Sequential Analysis, pages 503–529. Dekker, New York, NY, 1991.

[93] R. J. Serfling. *Approximation Theorems of Mathematical Statistics*. John Wiley & Sons, Inc., New York, NY., 1980.

[94] M. D. Serruya, N. G. Hatsopoulos, L. Paninski, M. R. Fellows, and J. P. Donoghue. Brain-machine interface: Instant neural control of a movement signal. *Nature*, 416(6877):141–142, 2002.

[95] Y. Shin, S. Lee, J. Lee, and H. Lee. Sparse representation-based classification scheme for motor imagery-based brain–computer interface systems. *Journal of Neural Engineering*, 9(5):056002, 2012.

[96] J. C. Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 34:817–823, 1998.

[97] J. C. Spall. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Transactions on Automatic Control*, 45:1839–1853, 2000.

[98] J. C. Spall. *Introduction to Stochastic Search and Optimization.* John Wiley & Sons, Inc., Hoboken, NJ., 2003.

[99] J. C. Spall. *Introduction to stochastic search and optimization: estimation, simulation, and control*, volume 65. John Wiley & Sons, 2005.

[100] J. C. Spall. Feedback and weighting mechanisms for improving Jacobian (Hessian) estimates in the adaptive simultaneous perturbation algorithm. In *Proceedings of the American Control Conference*, pages 35–40, June 2006.

[101] S. S. Sridhar and R. Shivaraman. Feasibility study for implementing brain computer interface using electroencephalograph. In *Proceedings of International Conference on Internet Computing and Information Communications*, pages 207–218. Springer, 2014.

[102] N. Starr. On the asymptotic efficiency of a sequential procedure for estimating the mean. *The Annals of Mathematical Statistics*, pages 1173–1185, 1966.

[103] N. Starr and M. B. Woodroofe. Remarks on sequential point estimation. *Proceedings of the National Academy of Sciences*, 63(2):285–288, 1969.

[104] S. Sun. The stochastic approximation method for learning adaptive bayesian classifiers: Towards on-line brain-computer interfaces.

[105] S. Sun, Y. Lu, and Y. Chen. The stochastic approximation method for adaptive bayesian classifiers: towards online brain–computer interfaces. *Neural Computing and Applications*, 20(1):31–40, 2011.

[106] S. Sun and C. Zhang. An optimal kernel feature extractor and its application to eeg signal classification. *Neurocomputing*, 69(13):1743–1748, 2006.

[107] K. Tavakolian and S. Rezaei. Classification of mental tasks using gaussian mixture bayesian network classifiers. In *Biomedical Circuits and Systems, 2004 IEEE International Workshop on*, pages S3–6. IEEE, 2004.

[108] A. Tayfun, M. Sun, R. J. Sclahassi, and A. Cetin. Characterization of sleep spindles using higher order statistics and spectra. *Biomedical Engineering, IEEE Transactions on*, 47(8):997–1009, 2000.

[109] D. M. Taylor, S. H. Tillery, and A. B. Schwartz. Direct cortical control of 3d neuroprosthetic devices. *Science*, 296(5574):1829–1832, 2002.

[110] M. Velliste, S. Perel, M. C. Spalding, A. S. Whitford, and A. B. Schwartz. Cortical control of a prosthetic arm for self-feeding. *Nature*, 453(7198):1098–1101, 2008.

[111] Y. Wang, B. Hong, X. Gao, and S. Gao. Implementation of a brain-computer interface based on three states of motor imagery. In *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, pages 5059–5062. IEEE, 2007.

[112] M. T. Wasan. *Stochastic Approximation*. Cambridge University Press, Cambridge, UK, 1969.

[113] C. Z. Wei. Multivariate adaptive stochastic approximation. *Annals of Statistics*, 15:1115–1130, 1987.

[114] J. Wessberg, C. R. Stambaugh, J. D. Kralik, P. D. Beck, M. Laubach, J. K. Chapin, J. Kim, S. J. Biggs, M. A. Srinivasan, and M. A. Nicolelis. Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. *Nature*, 408(6810):361–365, 2000.

[115] J. R. Wolpaw, N. Birbaumer, W. J. Heetderks, D. J. McFarland, P. H. Peckham, G. Schalk, E. Donchin, L. A. Quatrano, C. J. Robinson, and T. M. Vaughan. Brain-computer interface technology: a review of the first international meeting. *IEEE transactions on rehabilitation engineering*, 8(2):164–173, 2000.

[116] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan. Brain–computer interfaces for communication and control. *Clinical neurophysiology*, 113(6):767–791, 2002.

[117] F. Yousefian, A. Nedić, and U. V. Shanbhag. On stochastic gradient and subgradient methods with adaptive steplength sequences. *Automatica*, 45:56–67, 2011.

[118] F. Yousefian, A. Nedić, and U. V. Shanbhag. On stochastic gradient and subgradient methods with adaptive steplength sequences. *Automatica*, 48(1):56–67, 2012.

[119] S. Zhong and I. Ghosh. Hmms and coupled hmms for multi-channel eeg classification. In *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, volume 2, pages 1154–1159. IEEE, 2002.

Extended Numerical Results

LIAS Performance Under Different Posterior Probability Thresholds

Under varying posterior probability threshold (5.3), the performance of LIAS is evaluated. The following results show five restarts of LIAS, with probability threshold ranges from 0.6 to 0.9. The termination criterion is chosen to be reaching the maximum budget (i.e. size of the training set). As can be seen, although the performance is not quite sensitive to the choice of this parameter, the accuracy rate is slightly increasing, as the probability threshold is increasing. Meanwhile data utilization is decreasing, although not significantly.

In addition, a quick comparison between K-Means clustering results and LIAS's shows the effect of training via LIAS on the classifier's performance. Together with table of accuracy rates, also provided the sample size behavior; LIAS dictates dramatic increase in sample size as soon as the vicinity of a local minima is detected.

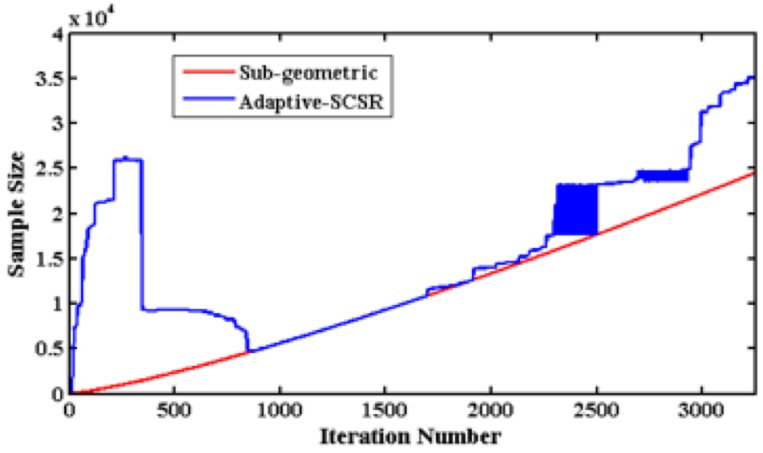

Figure 1: First run of LIAS on DS7: Sample Size Behavior

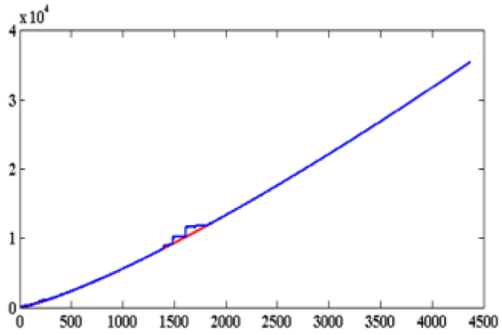

Figure 2: Second run of LIAS on DS7: Sample Size Behavior

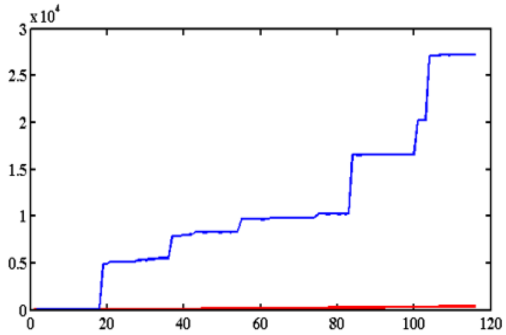

Figure 3: Third run of LIAS on DS7: Sample Size Behavior

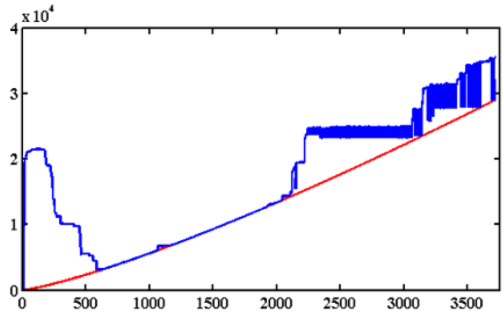

Figure 4: Fourth run of LIAS on DS7: Sample Size Behavior

	*Adaptive-SCSR parameter			*K-means parameter		
Threshold	Accuracy Rate	Data Utilization	Channel Capacity	Accuracy Rate	Data Utilization	Channel Capacity
0.6	0.685	0.577	0.421	0.342	0.471	0.340
0.62	0.712	0.505	0.373	0.315	0.397	0.289
0.64	0.740	0.439	0.329	0.356	0.334	0.240
0.66	0.740	0.376	0.282	0.356	0.282	0.202
0.68	0.740	0.319	0.239	0.329	0.234	0.170
0.7	0.753	0.265	0.201	0.315	0.195	0.142
0.72	0.767	0.221	0.169	0.329	0.162	0.117
0.74	0.740	0.181	0.136	0.342	0.133	0.096
0.76	0.795	0.145	0.113	0.384	0.109	0.077
0.78	0.740	0.116	0.087	0.384	0.088	0.062
0.8	0.767	0.091	0.070	0.397	0.070	0.050
0.82	0.767	0.070	0.054	0.425	0.056	0.039
0.84	0.767	0.055	0.042	0.397	0.043	0.030
0.86	0.753	0.042	0.032	0.384	0.033	0.023
0.88	0.740	0.032	0.024	0.384	0.024	0.017
0.9	0.740	0.024	0.018	0.397	0.018	0.013

Figure 5: First run of LIAS on DS7, under varying *Threshold*

Adaptive-SCSR parameter			K-means parameter		
Accuracy Rate	Data Utilization	Channel Capacity	Accuracy Rate	Data Utilization	Channel Capacity
0.493	0.930	0.650	0.507	0.976	0.683
0.493	0.915	0.640	0.507	0.972	0.679
0.493	0.901	0.630	0.507	0.968	0.676
0.493	0.887	0.620	0.507	0.963	0.673
0.493	0.871	0.609	0.507	0.957	0.669
0.493	0.856	0.598	0.507	0.951	0.665
0.521	0.838	0.586	0.507	0.946	0.661
0.507	0.822	0.575	0.507	0.941	0.657
0.507	0.803	0.561	0.507	0.933	0.652
0.521	0.783	0.548	0.507	0.926	0.647
0.521	0.763	0.534	0.507	0.917	0.641
0.521	0.742	0.519	0.507	0.909	0.635
0.534	0.716	0.501	0.507	0.899	0.629
0.521	0.690	0.482	0.507	0.889	0.622
0.534	0.661	0.462	0.507	0.876	0.612
0.534	0.628	0.439	0.507	0.861	0.602

Figure 6: Second run of LIAS on DS7, under varying *Threshold*

Adaptive-SCSR parameter			K-means parameter		
Accuracy Rate	Data Utilization	Channel Capacity	Accuracy Rate	Data Utilization	Channel Capacity
0.274	0.482	0.359	0.274	0.482	0.359
0.274	0.410	0.306	0.301	0.410	0.301
0.274	0.348	0.259	0.274	0.349	0.260
0.288	0.295	0.218	0.288	0.293	0.217
0.274	0.246	0.183	0.274	0.246	0.183
0.329	0.205	0.148	0.329	0.204	0.148
0.329	0.170	0.123	0.342	0.170	0.122
0.342	0.140	0.101	0.342	0.139	0.100
0.342	0.113	0.082	0.342	0.114	0.082
0.329	0.092	0.067	0.342	0.092	0.066
0.356	0.074	0.053	0.315	0.073	0.053
0.370	0.056	0.040	0.370	0.056	0.040
0.370	0.043	0.031	0.370	0.043	0.031
0.384	0.033	0.023	0.384	0.033	0.023
0.397	0.024	0.017	0.397	0.024	0.017
0.411	0.018	0.012	0.397	0.017	0.012

Figure 7: Third run of LIAS on DS7, under varying *Threshold*

Accuracy Rate	Data Utilization	Channel Capacity	Accuracy Rate	Data Utilization	Channel Capacity
0.712	0.596	0.441	0.288	0.484	0.358
0.726	0.526	0.392	0.301	0.409	0.300
0.767	0.457	0.349	0.301	0.343	0.252
0.753	0.392	0.297	0.315	0.289	0.211
0.753	0.333	0.252	0.342	0.241	0.174
0.753	0.280	0.212	0.288	0.204	0.151
0.753	0.232	0.176	0.301	0.170	0.125
0.781	0.189	0.145	0.329	0.140	0.102
0.767	0.154	0.118	0.315	0.114	0.083
0.795	0.125	0.098	0.342	0.091	0.066
0.781	0.099	0.077	0.370	0.072	0.051
0.753	0.078	0.059	0.397	0.057	0.041
0.740	0.060	0.045	0.425	0.045	0.032
0.767	0.046	0.035	0.384	0.034	0.024
0.699	0.035	0.026	0.411	0.025	0.017
0.699	0.025	0.018	0.411	0.018	0.012

Figure 8: Forth run of LIAS on DS7, under varying *Threshold*

Program Source

Line-Search Adaptive-SCSR (LIAS)

* DATE : Last updated August, 2015

* AUTHOR : Fatemeh S. Hashemi, Virginia Tech, Blacksburg, VA, fatemeh AT vt DOT edu

```
function [FinalAccuracy,FinalUtil,FinalChannelCap] = ...
    LIAS(Num_res,eta,scale_fn,Init_Policy,...
    Mixture_Policy,delta,K,N_c,csp_num,RanRes,Data_E,...
    N_trial_E,Data,N_trial,Lr)



% INPUT
%     Num_res
%         Number of random restarts
%     eta
%         Percentage of data with low likelihood, pruned as outliers
%     scale_fn
%         Functional scaler
%     Init_Policy
%         Parameter initialization policy
%     Mixture_Policy
%         1 implies adoption of equi-mixture GMM; otherwise implies
%         adoption of weighted mixture determined by K-Means.
%     delta
%         small positive constant to convert ND to PSD.
```

```
%     csp_num
%         Dimensionality of each signal after CSP.
%     K
%         Number of MI-tasks
%     N_c
%         Number of clusters in each Gaussian prototype.
%     RanRes
%         Random-restart activation code
%     Data_E
%         Testing data
%     N_trial_E
%         Number of trials in testing set
%     Data
%         Training data
%     N_trial
%         Number of trials in training set
%     Lr
%         Fixed step length in case line search is inactive
% OUTPUT
%     FinalAccuracy
%         Accuracy rate after multiple restart
%     FinalUtil
%         Data utilization after multiple restart
%     FinalChannelCap
%         Channel capacity after multiple restarts


usage=zeros(Num_res,1); % Data utilization on Training set
AccRate=zeros(Num_res,1); % Accuracy rate on Training set
Channel_cap=zeros(Num_res,1); % Channel Capacity on Training set
usage_E=zeros(Num_res,1); % Data utilization on Evaluating set
AccRate_E=zeros(Num_res,1); % Accuracy rate on Evaluating set
Channel_cap_E=zeros(Num_res,1); % Channel capacity on Evaluating set


for num_start=1:Num_res

    % Generate Initial Parameters :
    [Mu,Var,mix_prob]=InitPar(Data,Init_Policy,K,N_c,csp_num,RanRes);
    if Mixture_Policy==1
```

```
        mix_prob=(1/N_c)*ones(K,N_c);
        disp(mix_prob)
    end

    % Prune Data, and
    % derive maximum step length to be used in Line-search:
    [ Hm_H,Data_1]=GrandHess_DataPrune(Mu,Var,...
        mix_prob,Data,scale_fn,K,N_c,csp_num,eta);
    [max_s]=Max_StepLength(Hm_H,delta,K,N_c,csp_num);
    max_sl=max_s*200;
    if max_sl==0
        max_sl=4;
    end
    % Observe performance of LIAS over multiple restarts:
    [usage(num_start), AccRate(num_start), ...
        Channel_cap(num_start),usage_E(num_start),...
        AccRate_E(num_start), Channel_cap_E(num_start)]=...
        Classifier_LIAS(LS,c_1,beta_l,thresh,...
        ConvTolerance_Percentage,N_max,ConvRad,eps,c,max_sl,Mu,...
        Var,mix_prob,scale_fn,K,N_c,csp_num,Data_E,...
        N_trial_E,Data_1,N_trial,Lr);
end

% Final performance matrics:
FinalAccuracy=max(AccRate_E(AccRate==max(AccRate)));
FinalUtil=max(usage_E(AccRate==max(AccRate)));
FinalChannelCap=max(Channel_cap_E(AccRate==max(AccRate)));



function [Mu,Var,mix_prob]=InitPar(Data,Init_Policy,K,N_c,csp_num,RanRes)
%  generates initial patterns for LIAS classifier.

% INPUT
%    Data
%       Pre-processed EEG recordings.
%    Init_Policy
%       1 implies K-Means initialization; otherwise perturbed K-Means.
%    csp_num
%       Dimensionality of each signal after CSP.
```

```
%    K
%       Number of MI-tasks
%    N_c
%       Number of clusters in each Gaussian prototype.
% OUTPUT
%     MU
%       Updated decision parameter: mean vector for GMM
%     Var
%       Updated decision parameter: covariance matrix for GMM
%     mix_prob
%       Mixture probabilities in GMM.


if Init_Policy==1 % Initialization = K-Means solutions
    [Mu,Var,mix_prob] = K_means(Data,K,N_c,csp_num,RanRes);
else  % Initialization = Perturbed K-Means
    [Mu_center,Var,mix_prob] = K_Means(Data,K,N_c,csp_num,RanRes);
    Mu=Mu_center;
    for i=1:K
        for j=1:N_c
            Mu(i,j,:)=mvnrnd(reshape(Mu_center(i,j,:),csp_num,1),...
                reshape(Var(i,j,:,:),csp_num,csp_num));
        end
    end
end




function [Mu,Var,mix_prob] = K_Means(Data,K,N_c,csp_num,RanRes)
%  runs K-Mean clustering on Data.


% INPUT
%     Data
%         Data set
%     csp_num
%         Dimensionality of each signal after CSP.
%     K
%         Number of MI-tasks
```

```
%     N_c
%         Number of clusters in each Gaussian prototype.
%     RanRes
%         Random restart activation code

% OUTPUT
%    Mu
%       Decision parameter: Mean vector of GMM
%    Var
%       Covariance matrix of GMM
%    mix_prob
%       Mixture probabilities of GMM

Mu=zeros(K,N_c,csp_num);
Var=zeros(K, N_c,csp_num,csp_num);
mix_prob=zeros(K,N_c);
indx_fixed=[1 7700 12000];
classONE=Data(Data(:,csp_num+1)==1,1:csp_num);
s_1=size(classONE,1);

if RanRes==1
    generate_index1=round(1 + (length(classONE)-1).*rand(N_c,1));
else
    generate_index1=indx_fixed;
end


optns = statset('MaxIter',10000);

my_centers_1=classONE(generate_index1,:); % Initial soln for K-Means

[IDX1,mu_Class_1] = kmeans(classONE,N_c,'distance','cosine',...
    'start',   my_centers_1,'replicates',1, 'emptyaction',...
    'singleton', 'onlinephase','on', 'options',optns);

cov_1=zeros(N_c,csp_num,csp_num);
for i=1:N_c
    A=find(IDX1==i);
    mix_prob(1,i)=size(A,1)/s_1;
    B=classONE(A,:);
```

```
    cov_1(i,:,:)=cov(B);
end

Mu(1,:,:)=mu_Class_1;
Var(1,:,:,:)=cov_1;

classTWO=Data(Data(:,csp_num+1)==2,1:csp_num);
s_2=size(classTWO,1);

if RanRes==1
    generate_index2=round(1 + (length(classTWO)-1).*rand(1,N_c));
else
    generate_index2=indx_fixed;
end

my_centers_2=classTWO(generate_index2,:);

[IDX2,mu_Class_2] = kmeans(classTWO,N_c,'distance','cosine',...
    'start', my_centers_2, 'replicates',1, 'emptyaction',...
    'singleton', 'onlinephase','on', 'options',optns);


cov_2=zeros(N_c,csp_num,csp_num);
for i=1:N_c
    A=find(IDX2==i);
    mix_prob(2,i)=size(A,1)/s_2;
    B=classTWO(A,:);
    cov_2(i,:,:)=cov(B);
end

Mu(2,:,:)=mu_Class_2;
Var(2,:,:,:)=cov_2;



function [Hm_H,Data]=GrandHess_DataPrune(Mu,Var,...
    mix_prob,Data,scale_fn,K,N_c,csp_num,eta)

%  is pruning Data and calculating the grand
% Hessian estimator.
```

```
% INPUT
%    Data
%       Pre-processed EEG recordings.
%    Mu
%       Decision parameter: Mean vector of GMM
%    Var
%       Covariance matrix of GMM
%    mix_prob
%       Mixture probabilities of GMM
%    csp_num
%       Dimensionality of each signal after CSP.
%    K
%       Number of MI-tasks
%    N_c
%       Number of clusters in each Gaussian prototype.
%    scale_fn
%       Scale factor for the functionals
%    eta
%       Percentage of data pruned, with minimum likelihood
% OUTPUT
%    Hm_H
%       The true Hessian
%    Data
%       Pruned Data.



Hm_H=0;
l=length(Data);
Trashed_N=l*eta;
likelihood_val=zeros(l,1);

num=0;   % sample size
while num<l

    num=num+1;
    X=Data(num,1:csp_num);
    Type=Data(num,csp_num+1);
    p_xc = likelihood(X,Type, Mu,Var,mix_prob,K,N_c);
```

```
    likelihood_val(num)=p_xc(Type);
    [Hess_mu]= Hessian_Estimator(X,Type, Mu,Var,mix_prob,...
        csp_num,K,N_c,scale_fn);
    Hm_H=Hm_H+(Hess_mu);

end

[sort_ll,indx_sort]=sort(likelihood_val);
Data(indx_sort(1:Trashed_N),:)=[];




function [max_e]=Max_StepLength(Hess,delta,K,N_c,csp_num)
%  calculates the initial guess for step length in Linesearch.

%INPUT
%    csp_num
%       Dimensionality of each signal after CSP.
%    K
%       Number of MI-tasks
%    N_c
%       Number of clusters in each Gaussian prototype.
%    Hess
%       Input matrix
%    delta
%       Size of increment to convert ND to PSD matrix.
% OUTPUT
%    max_e
%       A measure for inverse maximum eigen value of the input matrix.
%

max_eig=zeros(K,N_c);

for i=1:K
    for j=1:N_c
        Hessian_ND=reshape(Hess(i,j,:,:),csp_num,csp_num);
        [R,p] = chol(Hessian_ND);
        if p~=0
            hess_inf=isinf(Hessian_ND);
```

```
            hess_NaN=isnan(Hessian_ND);
            detectINF=max(max(hess_inf));
            detectNaN=max(max(hess_NaN));
            if or(detectINF==1,detectNaN==1)
                error('Inf ot NaN Hessian observed!')
            else
                Hess_PSD=sqrtm(Hessian_ND'*Hessian_ND+...
                    (delta*eye(csp_num)));
                max_eig(i,j)=(1/max(eig(Hess_PSD)));
            end
        else
            max_eig(i,j)=(1/max(eig(Hessian_ND)));
        end
    end
end

max_e=max(max(max_eig));


function [Hess_mu] = Hessian_Estimator(X,Type, Mu,Var,...
    mix_prob,csp_num,K,N_c,scale_fn)
%  computes instant Hessian at X.

% INPUT
%    X
%       Data point, denoted by Z in Chapter 5.
%    Type
%       MI-tasks
%    Mu
%       Decision parameter: Mean vector of GMM
%    Var
%       Covariance matrix of GMM
%    mix_prob
%       Mixture probabilities of GMM
%    csp_num
%       Dimensionality of each signal after CSP.
%    K
%       Number of MI-tasks
%    N_c
%       Number of clusters in each Gaussian prototype.
```

```
%    scale_fn
%       Scale factor for the functionals
% OUTPUT
%    Hess_mu
%       instant Hessian


[p_xc,response] = likelihood(X,Type, Mu,Var,mix_prob,K,N_c);
if response==0
    Hess_mu=0;
else
    GMM=p_xc(Type);
    Fi2=zeros(K,N_c,csp_num,csp_num);

    for j=1:N_c   % cluster
        mu(:,1)=Mu(Type,j,:);
        VAR(:,:)=Var(Type,j,:,:);
        SigMu=(inv(VAR)*(X'-mu))*( inv(VAR)*(X'-mu))';
        G_2=mvnpdf(X',mu, VAR)*(SigMu-inv(VAR));
        Fi2(Type,j,:,:)=(mix_prob(Type,j))*(1/GMM)*G_2 - ...
            SigMu*((mix_prob(Type,j))*mvnpdf(X',mu, VAR)*(1/GMM))^2;
    end
    Hess_mu=-scale_fn*Fi2;
end



function [func_mu,Grad_mu,response] = Estimators(X,Type,...
    Mu,Var,mix_prob,csp_num,K,N_c,scale_fn)
%  computes instant function and gradient at X.

% INPUT
%    X
%       Data point, denoted by Z in Chapter 5.
%    Type
%       MI-tasks
%    Mu
%       Decision parameter: Mean vector of GMM
%    Var
%       Covariance matrix of GMM
```

```
%    mix_prob
%       Mixture probabilities of GMM
%    csp_num
%       Dimensionality of each signal after CSP.
%    K
%       Number of MI-tasks
%    N_c
%       Number of clusters in each Gaussian prototype.
%    scale_fn
%       Scale factor for the functionals
% OUTPUT
%    func_mu
%       instant function value
%    Grad_mu
%       Instant gradient value
%    response
%       1 implies X is responsive, otherwise it is unknown.


[p_xc,response] = likelihood(X,Type, Mu,Var,mix_prob,K,N_c);


if response==0
    func_mu=0;
    Grad_mu=0;
else
    GMM=p_xc(Type);
    Fi1=zeros(K,N_c,csp_num);

    for j=1:N_c
        mu(:,1)=Mu(Type,j,:);
        VAR(:,:)=Var(Type,j,:,:);
        G_1=mvnpdf(X',mu, VAR)* inv(VAR)*(X'-mu);
        Fi1(Type,j,:)=(mix_prob(Type,j))*G_1*(1/GMM);
    end
    func_mu=-scale_fn*(log(GMM)+log(1/K));
    Grad_mu=-scale_fn*Fi1;

end
```

```
function [usage, AccRate, Channel_cap,usage_E, AccRate_E,...
    Channel_cap_E] = Classifier_LIAS(LS,c_1,beta_l,thresh,...
    ConvTolerance_Percentage,N_max,ConvRad,eps,c,max_sl,Mu,...
    Var,mix_prob,scale_fn,K,N_c,csp_num,Data_E,N_trial_E,...
    Data,N_trial,Lr)
%  runs LIAS on a single start of SCSR.

% INPUT
%     LS
%        Step size policy; zero implies fixed step size policy;
%        otherwise, line search is active.
%     c_1
%        Back Tracking parameter (often chosen to be 0.0001)
%     beta_l
%        Line search parameter (often chosen to be 0.5)
%     thresh
%        Testing Probability Threshold (often chosen to be 0.65)
%     ConvTolerance_Percentage
%        Convergence tolerance percentage.
%     N_max
%        Max number of SCSR iterations allowed
%    ConvRad
%        Convergence radious
%    eps
%        coercion factor in Adaptive-SCSR stopping rule (e.g. 0.2)
%    c
%        balance factor in Adaptive-SCSR stopping rule (e.g. 1.02)

% OUTPUT
%    usage
%       Data utilization on training set
%    AccRate
%       Accuracy rate on training set
%    Channel_cap
%       Channel capacity on training set
%    usage_E
%       Data utilization on testing set
%    AccRate_E
%       Accuracy rate on testing set
```

```
%    Channel_cap_E
%       Channel capacity on testing set

alpha_k=zeros(N_max,1); % Step Size
Fn_MU=zeros(N_max,1); % true function value
XXX=zeros(K,N_c);  % std error of the gradient estimates
Func_MU=0; % sample function value
ConditionPass=1; % SCSR iteration number
nu=log(length(Data))/3; % escorting sequence is ConditionPass^nu
it_n=0; % number of SCSR iterations with max sample size
Run=1; % SCSR iterates while Run==1
Hm_mu=0; % sample gradient
num=0; % sample size

while Run==1;

    condition=0;  % Keep sampling.
    while condition==0

        num=num+1; % Increment sample size by one.
        % Update grand estimators:
        if num<=length(Data)
            X=Data(num,1:csp_num);
            Type=Data(num,csp_num+1);
            [func_mu,Grad_mu,resp0]= Estimators(X,Type,...
                Mu,Var,mix_prob,csp_num,K,N_c,scale_fn);
            Hm_mu=(((num-1)/num)*Hm_mu)+(Grad_mu/num);
            Func_MU=(((num-1)/num)*Func_MU)+(func_mu/num);
        end

        if num>1
            % Check the sampling rule:

            if num<=length(Data)
                XXX(:,:)=sqrt(((1/(num^2))*(sum((Hm_mu-Grad_mu).^2,3)))...
                    +   (((num-2)/num)*(XXX.^2))  );
                normHm(:,:)=sqrt(sum(Hm_mu.^2,3));
            end
            if or( (and(normHm>(c*XXX*(num^eps)),...
                    num>(ConditionPass+1)^nu)),(num>=length(Data)))
```

```
                condition=1; % Stop sampling.

                if num>length(Data)
                    LS=0;
                    % When the whole data is in use,
                    %line search becomes inactive;

                    Lr=alpha_k(ConditionPass-1);
                    num=length(Data);
                    it_n=it_n+1;
                    d_k=-Hm_mu*num;

                    [Mu,alpha_k(ConditionPass),Func_MU,Hm_mu,XXX] = ...
                        LineSearch(max_sl,d_k,LS,Lr,Func_MU,Hm_mu,...
                        Data, Mu,Var,mix_prob,csp_num,K,N_c,...
                        scale_fn,num,beta_l,c_1);

                    Fn_MU(it_n)=Func_MU;
                else
                    d_k=-Hm_mu*num;
                    [Mu,alpha_k(ConditionPass),Func_MU,Hm_mu,XXX] =...
                        LineSearch(max_sl,d_k,LS,Lr,Func_MU,...
                        Hm_mu,Data, Mu,Var,mix_prob,csp_num,K,N_c,...
                        scale_fn,num,beta_l,c_1);
                end
                ConditionPass=ConditionPass+1;  % update iteration number



                if it_n>ConvRad % Check termination criterion

                    [Run]=TerminCriterion(ConditionPass,N_max,...
                        ConvTolerance_Percentage,Fn_MU,it_n);
                end % if it_n>ConvRad
            end % if (stopping rule)
        end % if num>1
    end % while condition==0
end % while Run==1
```

```
% Test on the training set :
[usage, AccRate, Channel_cap] = testing_SCSR(scale_fn,Mu,Var,...
    mix_prob,thresh,N_trial,K,N_c,csp_num,Data);

% Test on the evaluation set
[usage_E, AccRate_E, Channel_cap_E] = testing_SCSR(scale_fn,...
    Mu,Var,mix_prob,thresh,N_trial_E,K,N_c,csp_num,Data_E);
```

```
function [Mu,alpha_k, New_fn, New_dk, New_SampleVar,...
    termin] = LineSearch(max_sl,d_k_1,LS,Lr,...
    Func_MU,Hm_mu_1,XXX_0,Data, Mu,Var,mix_prob,...
    csp_num,K,N_c,scale_fn,num,beta_l,c_1)

%  is Line Search procedure within LIAS.

% INPUT
%    max_sl
%       Initial step length for line search
%    d_k_1
%       Search Direction
%    LS
%       Activation code for Line-Search
%    Lr
%       Fixed step length in case Line search is inactive
%    Func_MU
%       Sample function at the curent solution
%    Hm_mu_1
%       Sample gradient at the current solution
%    XXX_0
%       Sample variance of the gradient estimate
%    Data
%       Pre-processed EEG recordings.
%    Mu
```

```
%       Decision parameter: Mean vector of GMM
%    Var
%       Covariance matrix of GMM
%    mix_prob
%       Mixture probabilities of GMM
%    csp_num
%       Dimensionality of each signal after CSP.
%    K
%       Number of MI-tasks
%    N_c
%       Number of clusters in each Gaussian prototype.
%    scale_fn
%       Scale factor for the functionals
%    num
%       Current sample size
%    beta_l
%       Shrinkage parameter in line search
%    c_1
%       Armijo rule constant
% OUTPUT
%     MU
%       Updated decision parameter
%     alpha_k
%       Step length
%     New_fn
%       Updated sample function
%     New_dk
%       Updated search direction
%     New_SampleVar
%       Updated sample variance of the gradient estimate
%     termin
%       "1" implies termination of SCSR; zero otherwise.


termin=0;
New_fn=0;
New_dk=0;
New_SampleVar=0;
Func_MU_u=0;
Hm_MU_u=0;
```

```
XXX=zeros(K,N_c);
if LS==0
    Mu_u=Mu+Lr*d_k_1;

    for nn=1:num
        X=Data(nn,1:csp_num);
        Type=Data(nn,csp_num+1);
        [func_mu_u,grad_u,resp0] = Estimators(X,...
            Type, Mu_u,Var,mix_prob,csp_num,K,N_c,scale_fn);
        if resp0==0
            Func_MU_u=Func_MU;
            Hm_MU_u=Hm_mu_1;
            XXX=XXX_0;
            termin=1;
            Mu_u=Mu;
            Lr=0;
            break
        end
        Func_MU_u=(((nn-1)/nn)*Func_MU_u)+(func_mu_u/nn);
        Hm_MU_u=(((nn-1)/nn)*Hm_MU_u)+(grad_u/nn);
        if nn>1
            XXX(:,:)=sqrt(( (1/(nn^2)) *...
                (sum((Hm_MU_u-grad_u).^2,3)))+(((nn-2)/nn)*(XXX.^2)));
        end
    end
    Mu=Mu_u;
    alpha_k=Lr;
    New_fn=Func_MU_u;
    New_dk=Hm_MU_u;
    New_SampleVar=XXX;
else

    d_k=zeros(K*N_c*csp_num,1);
    Hm_mu=zeros(K*N_c*csp_num,1);
    l=0;
    for i=1:K
        for j=1:N_c
            for h=1:csp_num
                l=l+1;
                d_k(l)=d_k_1(i,j,h);
```

```
            Hm_mu(l)=Hm_mu_1(i,j,h);
        end
    end
end
a_k=max_sl;
resp0=0;
condition_l=0; % When Wolf conditions are not yet satisfied, do:

while condition_l==0

    Mu_u=Mu+a_k*d_k_1;
    % calculate the updated state variable, with the new alpha_k.
    if a_k<1e-7
        % When a_k is too small, just return zero!
        condition_l=1; % alpha_k is found!
        Mu=Mu_u;
        % If Wolf conditions satisfied,
        % update the parameter calculated with the current alpha_k.
        alpha_k=a_k;
        New_fn=Func_MU_u;
        New_dk=Hm_MU_u;
        New_SampleVar=XXX;
        if resp0==0
            error('Line search failed!')
            % due to data rejection,
            % or the "very low maximum" step length.
        end
    else

        % Calculate the updated sample function
        % and sample gradient with the new alpha_k :
        Func_MU_u=0;
        Hm_MU_u=0;
        XXX=zeros(K,N_c);
        for nn=1:num
            X=Data(nn,1:csp_num);
            Type=Data(nn,csp_num+1);
            [func_mu_u,grad_u,resp0] = Estimators(X,...
                Type, Mu_u,Var,mix_prob,csp_num,K,N_c,scale_fn);
            if resp0==0
```

```
                    break
                end
                Func_MU_u=(((nn-1)/nn)*Func_MU_u)+(func_mu_u/nn);
                Hm_MU_u=(((nn-1)/nn)*Hm_MU_u)+(grad_u/nn);
                if nn>1
                    XXX(:,:)=sqrt(  ( (1/(nn^2)) *...
                        (sum((Hm_MU_u-grad_u).^2,3)))...
                        +   (((nn-2)/nn)*(XXX.^2)));
                end

            end

            if resp0==1
                if Func_MU_u<=(Func_MU+c_1*a_k*Hm_mu'*d_k)
                    New_fn=Func_MU_u;
                    New_dk=Hm_MU_u;
                    New_SampleVar=XXX;
                    condition_1=1; % alpha_k is found!
                    Mu=Mu_u;
                    % If Wolf conditions satisfied,
                    % update the parameters calculated
                    % with the current alpha_k.
                    alpha_k=a_k;
                end
            end
        end
        a_k=beta_1*a_k;    % scale down alpha_k;

    end
end



function [p_xc,response] = likelihood(X,Type, Mu,Var,mix_prob,K,N_c)
%  calculates the likelihood vector at input patterns.

% INPUT
%    X
%       Data point, denoted by Z in Chapter 5.
%    Type
```

```
%       MI-tasks
%    Mu
%       Decision parameter: Mean vector of GMM
%    Var
%       Covariance matrix of GMM
%    mix_prob
%       Mixture probabilities of GMM
%    K
%       Number of MI-tasks
%    N_c
%       Number of clusters in each Gaussian prototype.
% OUTPUT
%    p_xc
%       Vector of likelihood at X
%    response
%       1 implies X is responsive, otherwise it is unknown.

p_xc=zeros(K,1);
response=1;
for ii=1:K
    for j=1:N_c
        mu(:,1)=Mu(ii,j,:);
        VAR(:,:)=Var(ii,j,:,:);
        p_xc(ii)=p_xc(ii)+mix_prob(ii,j)*mvnpdf(X',mu, VAR);
    end
    if and(p_xc(ii)==0,ii==Type)
        response=0;
        break
    end
end


function [Run]=TerminCriterion(N_max,it_n,ConvRad,...
    ConvTolerance_Percentage,TrueFnVal)
%  checks the termination criterion for single start of LIAS.

% INPUT
%    ConvTolerance_Percentage
%        Convergence tolerance percentage.
%    N_max
```

```
%        Max number of SCSR iterations allowed
%    it_n
%        SCSR iteration number
%    ConvRad
%        Convergence radious
% OUTPUT
%    Run
%        SCSR termination code.

Run=1;

desig_error=zeros(ConvRad,1); % designated error

for i=1:ConvRad

  % Calculate percentage of change in TrueFnVal in successive rounds:
  desig_error(i)=100*(norm(TrueFnVal(it_n-i+1)- ...
      TrueFnVal(it_n-i))/norm(TrueFnVal(it_n-i)));

  % Designated error needs to be less than the convergence tolerance
  % within convergence radious:
  if desig_error(i)>ConvTolerance_Percentage
      break
  end
end
if or(i==ConvRad,it_n>N_max)
    Run=0;
end




function [usage, AccRate, Channel_cap] = ...
    testing_SCSR(Mu,Var,mix_prob,...
    thresh,N_trial,K,N_c,csp_num,Data)
%  produces the main performance metrics for LIAS classifier.

% INPUT
%    Data
%       Pre-processed EEG recordings.
```

```
%    Mu
%       Decision parameter: Mean vector of GMM
%    Var
%       Covariance matrix of GMM
%    mix_prob
%       Mixture probabilities of GMM
%    csp_num
%       Dimensionality of each signal after CSP.
%    K
%       Number of MI-tasks
%    N_c
%       Number of clusters in each Gaussian prototype.

% OUTPUT
%    usage
%       Data utilization
%    AccRate
%       Accuracy rate
%    Channel_cap
%       Channel capacity



class_decision=zeros(1,144);
report=0; % # responsive data
True=0; % # truly labeled trials
False=0; % # wrongly labeled trials

data_counter=0;
Init_trial=Data(1,csp_num+2);
Init_type=Data(1,csp_num+1);
Trial_num=Init_trial;
l=length(Data);
i=0; % Trial number
while data_counter<l
    i=i+1;
    class_1=0;  % # votes in favor of class 1
    class_2=0;  % # votes in favor of class 2

    % While in a single trial, do:
```

```
while and(Trial_num==Init_trial,data_counter<l)
    data_counter=data_counter+1;
    X=Data(data_counter,1:csp_num);
    Type=Data(data_counter,csp_num+1);
    Trial_num=Data(data_counter,csp_num+2);
    if Trial_num~=Init_trial
        Trial_type=Init_type;
        Init_type=Type;
        Init_trial=Trial_num;
        data_counter=data_counter-1;
        break
    end
    p_xc = likelihood(X,Type, Mu,Var,mix_prob,K,N_c);
    if sum(p_xc)~=0
        P_cx=p_xc/sum(p_xc);
        [Probability ,Class]=max(P_cx);
    else
        Probability=1/K;
        Class=0; % No decision can be made.
    end
    if Probability>thresh   && Class==1
        class_1=class_1+1;
        report=report+1;
    elseif Probability>thresh  && Class==2
        class_2=class_2+1;
        report=report+1;
    end

end
classes=[class_1 class_2];
[qq,class_decision(i)]=max(classes); % calculate majority vote

if qq~=0 % If votes exist, do:
    if  class_decision(i)==Trial_type
        True=True+1;
    elseif  class_decision(i)~=Trial_type
        False=False+1;
    end

end
```

```
end
AccRate=True/N_trial;
usage=(report/(length(Data)));
Channel_cap=usage*(AccRate*log2(AccRate)+1+(1-AccRate)*log2(1-AccRate));
```