ABSTRACT

Title of dissertation: RANDOMIZED SEARCH METHODS FOR SOLVING MARKOV DECISION PROCESSES AND GLOBAL OPTIMIZATION

Jiaqiao Hu, Doctor of Philosophy, 2006

Dissertation directed by: Professor Steven I. Marcus
Department of Electrical and Computer Engineering

Professor Michael C. Fu
Department of Decision and Information Technology

Markov decision process (MDP) models provide a unified framework for modeling and describing sequential decision making problems that arise in engineering, economics, and computer science. However, when the underlying problem is modeled by MDPs, there is a typical exponential growth in the size of the resultant MDP model with the size of the original problem, which makes practical solution of the MDP models intractable, especially for large problems. Moreover, for complex systems, it is often the case that some of the parameters of the MDP models cannot be obtained in a feasible way, but only simulation samples are available. In the first part of this thesis, we develop two sampling/simulation-based numerical algorithms to address the computational difficulties arising from these settings. The proposed algorithms have somewhat different emphasis: one algorithm focuses on MDPs with large state spaces but relatively small action spaces, and emphasizes on the efficient allocation of simulation samples to find good value function estimates, whereas the other algorithm targets problems with large action spaces but small state spaces, and invokes a population-based approach to avoid carrying out an optimization over the entire action space. We study the convergence properties of these algorithms and report on computational results to illustrate their performance.

The second part of this thesis is devoted to the development of a general framework called Model Reference Adaptive Search (MRAS) for solving global optimization problems. The method iteratively updates a parameterized probability distribution on the solution space, so that the sequence of candidate solutions generated from this distribution will converge asymptotically to the global optimum. We provide a particular instantiation of the framework and establish its convergence properties in both continuous and discrete domains. In addition, we explore the relationship between the recently proposed Cross-Entropy (CE) method and MRAS, and show that the model reference framework can also be used to describe the CE method and study its properties. Finally, we formally discuss the extension of the MRAS framework to stochastic optimization problems and carry out numerical experiments to investigate the performance of the method.

RANDOMIZED SEARCH METHODS FOR SOLVING MARKOV DECISION PROCESSES AND GLOBAL OPTIMIZATION

by

Jiaqiao Hu

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2006

ACKNOWLEDGMENTS

I would like to express my sincerest thanks to my advisors Professor Steven I. Marcus and Professor Michael C. Fu, whose consistent guidance and encouragement have made my five years of graduate studies an unforgettable and exciting experience. Not only have they provided generous financial support to me, but they have also given me a valuable opportunity to work on different interesting problems, as well as numerous suggestions and advices in my research and vocational matters. I am also grateful to Hyeong Soo Chang at the Sogang University, Seoul, Korea for his collaboration. Part of this research was inspired by his insightful ideas.

I would also like to thank the other members of my dissertation committee, Professor P. S. Krishnaprasad, Professor Dana Nau, and Professor André Tits for serving on my thesis committee and sparing their valuable time reviewing the manuscript.

I do not have enough words to express my thanks to my parents and my wife, for their patience and enduring support. Without them, this dissertation would not have been possible.

This research was supported in part by the National Science Foundation under Grants DMI-9988867 and DMI-0323220, and by the Air Force Office of Scientific Research under Grants F496200110161 and FA95500410210.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Markov Decision Processes

Markov Decision Processes (MDPs) are widely used for modeling and describing sequence decision making under uncertainty that arises in various areas such as manufacturing systems, financial engineering, artificial intelligence, and operations research. An MDP model consists of four principal components: a state space, an action space, the effects of the actions and the immediate cost incurred by the actions. The relations among these components are illustrated as follows.

Consider a decision maker that interacts simultaneously with his environment over a finite or infinite time horizon divided into a sequence of stages (decision epochs). At each stage, the decision maker observes the *state* of the environment, where it is assumed that the observation is complete and perfect; based on his observation, a *decision* (an *action*) is made to react to the environment. The decision influences (either deterministically or stochastically) the state at the next stage, and depending on the state and the decision made, a certain *cost* is incurred. The expected total costs accumulated from the current stage to the end of the planning horizon is called a *value function.* The goal of the decision maker is to find a *decision rule/policy* specifying the best action to take for each of the states, so that he can act optimally with the changing environment, in the sense that the expected total (discounted) cost over the entire planning horizon is minimized.

In finite horizon problems, the optimal decision rules (policies) generally depend on both the stage and state; they can be computed by the classical dynamic programming

(DP) algorithm starting from the terminal stage. In DP, the optimal decisions are determined backwards step by step as the minimizers of a functional equation, which expresses the value function at the present stage as the sum of the one-stage current cost and the value function at the following stage. This way of determining the optimal policy is based on Bellman's principle of optimality, which says, "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision" (cf. [63]).

There are a variety of solution methods for solving infinite horizon MDPs, many of which can be viewed as different strategies for solving Bellman's equation. The two most well-known approaches are value iteration (VI) and policy iteration (PI). Value iteration is essentially the extension of the DP algorithm to the infinite horizon case; it starts with an arbitrary (bounded) function and updates at each iteration the current function into a new function that better approximates the optimal value function. Thus the algorithm essentially amounts to using the solution to a finite but large horizon problem to approximate the solution to the infinite horizon problem. As an alternative to VI, policy iteration starts with an arbitrarily chosen stationary policy and generates a sequence of new policies. At each iteration of PI, a policy evaluation step is carried out to compute the value function associated with the current policy as the solution of a system of linear equations. Once this value function is obtained, a policy improvement step is used to generate a new policy that improves the performance (in terms of value function) of the current one. The process is repeated until no further improvement can be achieved.

There are also various straightforward enhancements of VI and PI for solving MDPs, including the methods that reduce the computational cost of VI and PI by directly applying the standard iterative schemes for solving systems of linear equations such as the

Gauss-Seidel method (cf. e.g., [13] and [63]) and the successive over relaxation (SOR) method ([81]). Puterman and Shin [62] proposed a modified policy iteration algorithm, which takes the basic form of PI, with the difference being that the policy evaluation step is carried out only approximately by executing a limited number of value iteration steps. The algorithm combines the advantages of VI and PI, and thus to some extent, alleviates the high computational burden via directly (e.g., Gaussian elimination) solving systems of linear equations (i.e., Bellman's equation).

For the sake of completeness, it is worth mentioning that the linear programming (LP) approach has also long been established as a useful method for solving infinite horizon discounted cost MDPs (cf. [13], [63]). The basic idea of the LP approach is to formulate the Bellman's equation as a set of linear constraints over all state-action pairs and interpret the optimal value function as the "largest" (in a minimization context) value function that satisfies these constraints.

The aforementioned approaches may quickly lead to computational intractability, since they require enumerating the entire state and action spaces, which often grow exponentially fast with the parameters of the problem (i.e., the well-known "curse of dimensionality"). In order to address this issue, many researchers have used various approximation schemes to reduce the size of the state/action spaces.

1.1.1 State Space Reduction Techniques

Bertsekas and Castañon [15] proposed a class of adaptive aggregation algorithms for solving infinite horizon MDPs. The idea is to group the states of the original problem into a smaller number of aggregate states in such a way that the resulting aggregated states actually constitute a smaller MDP. If the size of the resultant problem is small enough, then its value function can be computed exactly by directly solving the system of linear equations. The value function is in turn used to approximate the value function of the original problem by using some deaggregation schemes.

Unlike the state aggregation approach, some other approaches have concentrated on approximating the value function via a suitable parameterization, in effect restricting the search to a smaller-dimensional parameter space instead of the entire state space. The approximation is carried out via a number of different techniques: Bellman et al. [11] explored the use of polynomial approximations as compact representations of the value function in order to accelerate dynamic programming. Schweitzer and Seidmann [73] developed several techniques for approximating value functions using linear combinations of fixed sets of basis functions. More recently, Tsitsiklis and Van Roy [83] developed algorithms that employ the feature-based compact representations of the value function in dynamic programming. One of their algorithms was successfully applied to play the Tetris game. Trick and Zin [82] studied approaches based on linear programming for solving large MDPs and considered the use of low-dimensional cubic-spline approximations to the value function. In De Farias and Van Roy [27], the value function was approximated by a linear combination of pre-selected basis functions. The approach was used in conjunction with linear programming for approximately solving infinite horizon discounted cost problems.

Another class of methods explores the use of Monte Carlo integration to avoid the high computational cost of multivariate numerical integration that appears in the value iteration approach. The most notable work in this area is due to Rust [72], who used a randomized version of the Bellman operator to solve a class of MDPs with finite action spaces called the discrete decision processes (DDP). Rust showed that (under some regularity conditions) the amount of computational time required for his algorithms to solve the DDP problem increases only polynomially rather than exponentially with the dimension of the state variables.

All the computational methods mentioned so far require an explicit, complete mathematical model of the system to be controlled, represented by the availability of the cost structure and the transition probabilities. There is a class of methods, on the other hand, does not require the explicit specification of the transition probabilities and one-stage costs. Instead, they rely on the use of Monte Carlo simulation methods, where the underlying system can be simulated. In the artificial intelligence community, these approaches are often referred to as reinforcement learning, which include the method of temporal difference ([80]) and Q-learning ([85]), as well as certain variations and extensions of them (cf. e.g., [13] for a review). Recently, there have been some new and exciting ideas that combine the use of the specialized MDP techniques with the solution strategies in the area of global optimization. In these approaches, the simulation techniques are used not only to resolve the issue of the unavailability of the explicit parameters of MDP models, but also to avoid searching (enumerating) the entire (large or uncountable) state or solution space. Chang et al. [20] proposed an algorithm based on the idea of simulated annealing ([50]) for solving finite horizon MDPs. The algorithm works directly on the policy space and iteratively updates a probability distribution over a given set of policies. They showed that the sequence of distributions will converge to a distribution concentrated only on the optimal policies. A similar but more general framework was also proposed in [58], where MDPs with several reward (cost) criteria are formulated as global optimization problems over the set of all admissible policies, and are thus solved by using the cross-entropy (CE) method ([26], [66], [67], [68]). The efficiency of their approach is demonstrated for an inventory control problem and a maze problem.

1.1.2 Action Space Reduction Techniques

In contrast to large state spaces, the issue of large action spaces has been much less explored. It was partially addressed in early work by MacQueen [57], who used some inequality forms of Bellman's equation together with bounds on the optimal value function to identify and eliminate non-optimal actions in order to reduce the size of the action sets to be searched at each iteration of the algorithm. Since then, the procedure has been applied to several standard methods like policy iteration (PI), value iteration (VI) and modified policy iteration (cf. e.g., [63] for a review). In a recent paper [30], the action elimination idea has been explored in a reinforcement learning context where the explicit MDP model is not known. So far, all of these algorithms generally require that the admissible set of actions at each state is finite.

1.2 Global Optimization

The goal of global optimization is to find parameter values that achieve the optimum of an objective function. In general, due to the presence of multiple local optimal solutions, global optimization problems are typically extremely difficult to solve exactly. This section briefly reviews some of the standard global optimization algorithms with an emphasis on general solution techniques that are applicable to both combinatorial and continuous optimization problems.

Methods for global optimization can be categorized based on a number of different criteria. For instance, they can be classified either based on the properties of problems to be solved (combinatorial or continuous, nonlinear, linear, convex, etc.) or by the types of guarantees that the methods provide for the final solution. The classification that best fits our proposed research is from the algorithmic point of view, where solution algorithms are categorized as being either *instance-based* or *model-based*; cf. [91].

1.2.1 Instance-based Methods

In *instance-based* methods, the searches for new candidate solutions depend explicitly on previously generated solutions. Some well-known approaches are simulated annealing (SA) ([50]), genetic algorithms (GAs) ([79]), tabu search ([35]), and the recently proposed nested partitions (NP) method ([75], [76]).

Simulated annealing was initially introduced to solve combinatorial optimization problems. The algorithm starts out with some initial configuration/solution, and the neighbors (candidate solutions) of the current solution are randomly visited. The key idea of the algorithm is that neighbors that are either better or worse than the current solution may both be accepted with a certain probability, and the probability of accepting worse solutions gradually decreases during the search process. Thus the technique gives a simple local search algorithm the possibility to escape from local optimal solutions. The algorithm was later extended to solve continuous optimization problems by Corana et al. [24].

Genetic algorithms are inspired by natural selection and survival of the fittest in the biological world. In GAs, a population rather than a single solution is considered. Each iteration of the algorithm involves a "crossover" and a "mutation", where promising solutions are recombined with other solutions by swapping parts of a solution with another, and are then "mutated" by making a small change to the solution. The rationale is that recombination and mutation may give rise to new solutions that are biased towards regions containing good solutions.

The basic idea of tabu search is to record the search process, so that a search path already visited can be avoided. This insures new regions of the solution space will be investigated with the goal of avoiding local minima and ultimately finding the desired solution.

The nested partitions method systematically partitions the solution space into smaller subregions, accesses the potential of each region based on random sampling, and concentrates the computational efforts in the most promising region. This is done repeatedly until some of the regions are singleton sets (i.e., containing only one solution). In some sense, this is equivalent to changing the underlying sampling distribution in that more promising solutions will have larger chances of being selected. The algorithm is shown to converge to a global optimal solution with probability one.

1.2.2 Model-based Methods

The *model-based* search methods are a class of new solution techniques and were introduced only in recent years. In *model-based* algorithms, new solutions are generated via an intermediate probabilistic model that is updated or induced from the previously generated solutions. So there is only an implicit/indirect dependency among the solutions generated as successive iterations of the algorithm. In general, most of the algorithms that fall in this category share a similar framework and usually involve the following two phases:

1. Generate candidate solutions (random samples, trajectories) according to a specified probabilistic model (e.g., a parameterized probability distribution on the solution space).

2. Update the probabilistic model, on the basis of the data collected in the previous step, in order to bias the future search toward "better" solutions.

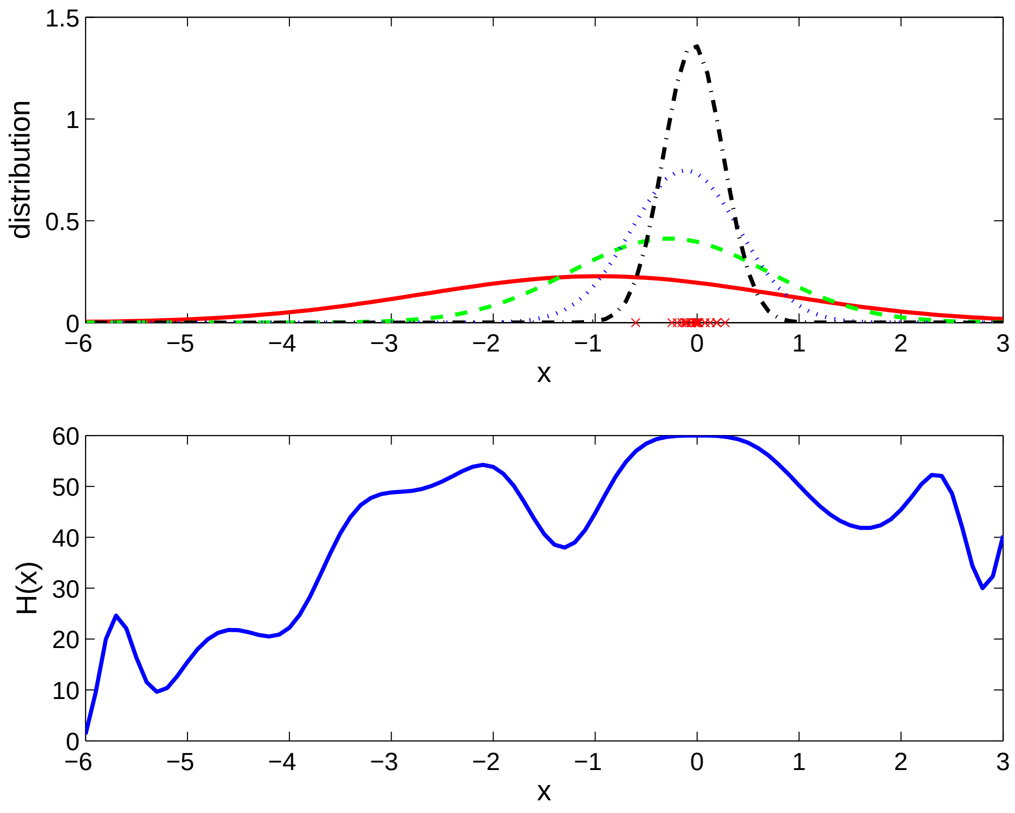


Figure 1.1: Optimization via model-based methods

To illustrate how model-based methods work, we consider, in Figure 1.1, maximizing a one-dimensional multi-extremal function $H(x)$, where its global optimum is achieved at $x = 0$. The model-based methods approach this problem by initially casting a probability model (distribution) over the solution space (the solid curve in Figure 1.1). This initial distribution is then used to generate candidate solutions/samples, the performance of these samples are evaluated and are thus used to update the initial distribution to obtain a new distribution (the dashed curve in the figure). The preceding procedure is performed repeatedly until some stopping criteria is satisfied. The underlying idea is that if these probabilistic models are updated in an appropriate way, then the sequence of samples/candidate solutions generated will become more and more concentrated near the optimum.

Some well established techniques that belong to the *model-based* methods are the cross-entropy (CE) method ([26],[58],[65],[66],[67],[68]), a class of algorithms called the estimation of distribution algorithms (EDAs) ([53],[59],[60]), and the so-called annealing adaptive search (AAS) ([74],[89]). The CE method was motivated by an adaptive algorithm for estimating probabilities of rare events in complex stochastic networks ([65]), which involves variance minimization. It was soon realized ([66], [67]) that the method can be modified to solve combinatorial and continuous optimization problems. The CE method usually starts with a family of parameterized probability distributions on the solution space and tries to find the parameter of the distribution that assigns maximum probability to the set of optimal solutions. Implicit in CE is an optimal reference distribution concentrated only on the set of optimal solutions (i.e., zero variance). The key idea of CE is to use an iterative scheme to successively estimate the optimal parameter that minimizes the KL-divergence between the optimal reference distribution and the family of parameterized distributions. The literature analyzing the convergence properties of the CE method is relatively sparse. In the context of estimation of rare event probabilities, Homem-de-Mello ([41]) shows the convergence of a variational version of CE to an estimate of the optimal (possibly local) CE parameter with probability one. Rubinstein ([66]) shows the probability one convergence of the CE method to the optimal solution for combinatorial optimization problems.

The estimation of distribution algorithm (EDA) was first introduced in the field of evolutionary computation in [59]. It inherits the spirit of the well-known genetic algorithms (GAs), but eliminates the crossover and the mutation operators in order to avoid the disruption of partial solutions. In EDAs, a new population of candidate solutions are generated according to the probability distribution induced or estimated from the promising solutions selected from the previous generation. Unlike CE, EDA often takes into account the interrelations between the underlying decision variables needed to represent the individual candidate solutions. At each iteration of the algorithm, a high-dimensional probabilistic model that better represent the interdependencies between the decision variables is induced; this step constitutes the most crucial and difficult part of the method. We refer the reader to [53] for a review of the way in which different probabilistic models are used as EDAs instantiations. The convergence of a class of EDAs, under the infinite population assumption, to the global optimum can be found in [90].

In annealing adaptive search (AAS) (cf., e.g., [89]), there is a sequence of distributions called Boltzmann distributions, each is parameterized by a temperature parameter T. One salient property of the Boltzmann distribution is that when T decreases to 0, the sequence of Boltzmann distributions will converge to a degenerated distribution concentrated only on the optimum. So the idea behind AAS is that if we can repeatedly sample from the Boltzmann distribution as the temperature parameter gradually decreases to 0, then the candidate solutions/samples generated will converge to the global optimum. However, sampling from the Boltzmann distribution is extremely difficult if not possible, since the distribution depends on the objective function itself. Thus in AAS, the research and computational efforts have mostly centered around the issue of how to efficiently generate samples. Currently, one popular and successful sampling approach is via the use of Markov Chain Monte Carlo (MCMC) [89], but the distribution of the samples generated according to MCMC can only be guaranteed to converge to the true Boltzmann distribution in an asymptotic sense ([89]).

1.3 Research Contributions

The main contributions of this thesis are as follows:

- We have developed a simulation-based multistage sampling algorithm for solving finite horizon MDPs. The algorithm is motivated by the computational challenges arising from settings where some of the parameters of the MDP models are either unknown or cannot be obtained in a feasible way. We have assumed that the underlying system can be simulated and proposed to use multi-armed bandit models as efficient tools to capture the tradeoff between sampling a promising action repeatedly and exploring further other actions that might yield even greater benefit, so that computational resources can be efficiently allocated in an adaptive manner as the sampling process proceeds. We have studied the convergence properties (including rate and complexity) of the algorithm and reported on computational results to illustrate its performance. This work has been published in *Operations Research* [22].

- Our second contribution complements those aforementioned state space reduction techniques (cf. Section 1.1.1) and focuses on the issue of large action spaces. In particular, we have proposed a novel algorithm that uses evolutionary, population-based approaches to directly searching the policy space in order to avoid carrying out an optimization over the entire action space. We have established the convergence of the algorithm for MDPs with finite state space but general (Borel) action spaces and compared the performance of the algorithm with those of the existing techniques. Preliminary empirical results on a queueing example indicated that the proposed method may significantly reduce the computational effort of the classical PI algorithm. A slightly different version of this work has been accepted for publication at *INFORMS Journal on Computing* [45].

- We have also proposed a new general framework called Model Reference Adaptive Search (MRAS) for solving global optimization problems, which addresses the most common computational difficulties faced by many model-based methods. We have provided a particular instantiation of the framework and analyzed its global convergence properties. We have studied some of the important properties of the recently proposed CE method and showed that the CE method can actually be interpreted as an instance of the proposed framework. We have also carried out detailed numerical studies to demonstrate the effectiveness of the method and compared its performance with those of CE and SA. This work has been accepted for publication at *Operations Research* [46]; a preliminary version of this work was presented at the 2005 Genetic and Evolutionary Computation Conference (GECCO) [43].

- We have extended the MRAS framework to stochastic global optimization problems, derived a set of sufficient conditions to ensure the global convergence of the method, and tested the approach on several benchmark problems such as (s,S) inventory control problem and optimal buffer allocation problems in unreliable production lines. This work has been submitted for publication [47]; a much abbreviated version appeared in the 2005 Winter Simulation Conference proceedings [44].

The rest of this thesis is structured as follows.

Chapter 2 provides some necessary background on MDPs and global optimization. Specifically, Chapter 2.1 gives the formal definition of the MDP model and presents the two classical approaches, value iteration (VI) and policy iteration (PI), for solving the model. Chapter 2.2 briefly describes two of the recently proposed *model-based* methods for solving global optimization with an emphasis on the cross-entropy (CE) method, which will be our starting points for deriving results of Chapter 5 and Chapter 6.

In Chapter 3, we introduce a simulation-based algorithm called Adaptive Multi-stage Sampling (AMS) for solving finite horizon MDPs with finite state and action spaces. The algorithmic procedure is described in Chapter 3.2. The detailed convergence analysis is given in Chapter 3.3. In Chapter 3.4, we perform computational experiments on a set of inventory control problems, provide two additional estimators, and discuss the performance of different estimators.

In Chapter 4, we propose a novel algorithm for solving a class of problems where the state space is relatively small but the action space is large or uncountable. The chapter contains a detailed description of the proposed algorithm in Chapter 4.3, a theoretical convergence proof of the algorithm in Chapter 4.4, and some preliminary empirical results in Chapter 4.6. Along the discussion, an adaptive version of the proposed algorithm is also considered and discussed in Chapter 4.5.

In Chapter 5, we propose a new *model-based* framework for solving global optimization. A specific instantiation of the framework, in its deterministic version, as well as its convergence properties, are presented and established in Chapter 5.3, whereas the corresponding Monte Carlo version of the method is described and its convergence proved in Chapter 5.5. We explore the relationship between the CE method and the proposed framework in Chapter 5.4. Preliminary numerical studies are also carried out in Chapter 5.6 to demonstrate the effectiveness of the method.

Chapter 6 summarizes our initial idea in adapting the MRAS framework to stochastic domains. In particular, we provide a variational extension of the MRAS method in Chapter 6.3, prove its global convergence in Chapter 6.4, and carry out numerical experiments in Chapter 6.5 to verify the theoretical findings.

Finally, we conclude the thesis in Chapter 7 with a summary of the work done, a discussion of the unresolved open issues, and an outline of some possible future research topics.

Chapter 2

Preliminaries

2.1 Markov Decision Processes

The MDP model can be formally described by a five-tuple $M = (X, A, \{P_t,\ t = 0, 1, \ldots\}, \{R_t,\ t = 0, 1, \ldots\}, \alpha)$, where

- X is a finite set of states of the environment.
- A is a general action space.
- $\{P_t,\ t = 0, 1, \ldots\}$ is a sequence of state transition matrices, each maps a state-action pair to a probability distribution over the state space X. At time t, the probability of transitioning to state $y \in X$, given that we are in state $x \in X$ taking action $a \in A$, is denoted by $P_{x,y|t}(a)$, i.e., the (x, y)th entry of P_t.
- $\{R_t,\ t = 0, 1, \ldots\}$ is a sequence of bounded non-negative one-stage cost functions, where at time t, $R_t : X \times A \to \Re^+ \cup \{0\}$.
- $\alpha \in (0, 1]$ is a discount factor.

Let $x_t,\ t = 0, 1, \ldots$, a random variable taking its values in X, be the state of the system at time t. A decision rule or policy is a sequence of functions $\boldsymbol{\pi} := \{\pi_t,\ t = 0, 1, \ldots\}$ with each $\pi_t : X \to A$ specifying the action $\pi_t(x)$ taken when in state $x_t = x \in X$ at time t. Such a policy is called stationary if all its components are independent of time, i.e., it takes the form $\boldsymbol{\pi} := \{\pi, \pi, \ldots\}$; for notational brevity, we simply denote it by π.

For a given horizon length $T > 0$, a given policy $\boldsymbol{\pi} = \{\pi_t,\ t = 0, 1, \ldots, T-1\}$ and an initial state x_0, a particular system path that the decision maker follows is given by a sequence of states and actions $\{x_0, \pi_0(x_0), \ldots, x_t, \pi_t(x_t), x_{t+1}, \pi_{t+1}(x_{t+1}), \ldots\}$, where the transitioning from x_t to x_{t+1} is determined by the probability $P_{x_t, x_{t+1}|t}(\pi_t(x_t))$. Thus, the probability of taking this particular path can be calculated as $\prod_{t=0}^{T-1} P_{x_t, x_{t+1}|t}(\pi_t(x_t))$, and the corresponding accumulated total cost can also be expressed as $\sum_{t=0}^{T-1} \alpha^t R_t(x_t, \pi_t(x_t))$. Thus, under the discounted cost criterion, which will be the primary focus of this research, the expected total accumulated cost over all possible sample paths associated with $\boldsymbol{\pi}$ can be expressed as

$$J^{\boldsymbol{\pi}}(x) = E\left[R_T(x_T) + \sum_{t=0}^{T-1} \alpha^t R_t(x_t, \pi_t(x_t)) \Big| \, x_0 = x \right],\ x \in X,\ \alpha \in (0, 1].$$

If the horizon length $T = \infty$, we assume that both the transition probability P and the one-stage cost function R are stationary, i.e., they do not change with time t. We therefore drop the explicit display of t in both P and R, and write the expected total discounted cost over an infinite horizon as

$$J^{\boldsymbol{\pi}}(x) = E\left[\sum_{t=0}^{\infty} \alpha^t R(x_t, \pi_t(x_t)) \Big| \, x_0 = x \right],\ x \in X,\ \alpha \in (0, 1),$$

where note that we require α to be strictly less than 1 in this case.

In both cases, we let $J^*(x)$ be the optimal cost function starting with an initial state x, defined by

$$J^*(x) = \inf_{\boldsymbol{\pi}} J^{\boldsymbol{\pi}}(x), \quad x \in X. \tag{2.1}$$

We also call a stationary policy π optimal if $J^{\pi}(x) = J^*(x)\ \forall\ x \in X$.

For finite horizon problems, i.e., $T < \infty$, it is well-known that the optimal cost $J^*(x)$ can be obtained via the following recursion (cf. e.g., [13] Vol. II).

Theorem 2.1.1 *For every initial state $x \in X$, the optimal cost $J^*(x)$ is equal to $J_0(x)$, given by the last step of the following algorithm, which proceeds backward in time from stage $T-1$ to stage 0:*

$$\begin{aligned} J_T(x) &= R_T(x), \ \forall x \in X \\ J_t(x) &= \min_{a_t \in A} \left[R_t(x, a_t) + \alpha \sum_{y \in X} P_{x,y|t}(a_t) J_{t+1}(y) \right], \ \forall x \in X, t = 0, \ldots, T-1. \end{aligned} \tag{2.2}$$

Furthermore, if $a_t^* = \pi_t^*(x)$ minimizes the right hand side of equation (2.2) for each x and t, the policy $\boldsymbol{\pi}^* = \{\pi_0^*, \ldots, \pi_{T-1}^*\}$ is optimal.

For infinite horizon problems, i.e., $T = \infty$, the optimal cost function J^* satisfies the following Bellman's optimality equation, which is essentially a stationary counterpart of equation (2.2).

Theorem 2.1.2 *Under the bounded cost assumption and $\alpha \in (0, 1)$, the optimal cost J^* satisfies*

$$J^*(x) = \min_{a \in A} \left[R(x, a) + \alpha \sum_{y \in X} P_{x,y}(a) J^*(y) \right]. \tag{2.3}$$

Note that for simplicity, we have assumed in equations (2.2) and (2.3) that all actions in A are admissible for each state in X.

The following proposition implies the existence of a stationary optimal policy when the minimum in the right hand side of Bellman's equation is attained for all $x \in X$.

Proposition 2.1.1 *A stationary policy π is optimal if and only if $\pi(x)$ attains the minimum in Bellman's equation (2.3) for all $x \in X$.*

Note that when the action space A is finite, a stationary optimal policy is guaranteed to exist. On the other hand, when A is infinite, we can also ensure the existence of such a policy by imposing some regularity assumptions on A, P, and R such that the minimum in equation (2.3) is attained. For ease of exposition, we will simply assume that a stationary optimal policy for problem (2.1) always exists under the infinite horizon setting.

We now briefly describe the two most basic approaches for solving Bellman's equation in an infinite horizon setting: value iteration (VI) and policy iteration (PI). Their detailed discussions can be found in [13] and [63].

2.1.1 Value Iteration

VI is basically the dynamic programming (DP) algorithm and is a principal method for computing the optimal value function J^*. It starts with an arbitrary bounded function $J_0(x)\ \forall x \in X$, and computes at each iteration $k = 0, 1, \ldots$ a new function $J_{k+1}(x)\ \forall\ x \in X$ from the old function $J_k(x)$ according to

$$J_{k+1}(x) = \min_{a \in A} \left[R(x,a) + \alpha \sum_{y \in X} P_{x,y}(a) J_k(y) \right], \ \forall\ x \in X. \tag{2.4}$$

It is well-known that under some mild regularity assumptions, the sequence of functions $\{J_k,\ k = 0, 1, \ldots\}$ generated will converge to the optimal value function, i.e., $\lim_{k\to\infty} J_k(x) = J^*(x)\quad \forall\ x \in X$ (cf. e.g., [13] and [63]). VI will generally require infinite number of iterations to compute the optimal value function; however, in practice, the algorithm can often be strengthened by the use of some error bounds. It can be shown (cf. [13] and [63]) that for a predetermined tolerance $\varepsilon > 0$, if $|J_{k+1}(x) - J_k(x)| < \varepsilon\ \forall\ x \in X$ for some k, then the value function corresponding to the greedy policy π^k that attains the minimum in the kth iteration of equation (2.4) can not be too "far away" from the optimal value function J^*, in the sense that

$$\max_{x \in X} |J^{\pi^k}(x) - J^*(x)| < 2\varepsilon \frac{\alpha}{1-\alpha}.$$

The above error bounds often provide a useful guideline for terminating the VI algorithm.

2.1.2 Policy Iteration

As an alternative to VI, PI starts with an arbitrary initial stationary policy π_0 and generates, one at each iteration, a sequence of stationary policies $\{\pi^0, \pi^1, \pi^2, \ldots\}$. At each iteration $k = 0, 1, \ldots$, the following two steps are fundamental:

1. Policy evaluation step that evaluates the value function J^{π^k} associated with the current (stationary) policy π^k.

$$J^{\pi^k}(x) = R(x, \pi^k(x)) + \alpha \sum_{y \in X} P_{x,y}(\pi^k(x)) J^{\pi^k}(y), \quad \forall\, x \in X. \tag{2.5}$$

2. Policy improvement step, which computes a new improved policy π^{k+1} as

$$\pi^{k+1}(x) = \arg\min_{a \in A} \left[R(x, a) + \alpha \sum_{y \in X} P_{x,y}(a) J^{\pi^k}(y) \right], \quad \forall\, x \in X. \tag{2.6}$$

It can be shown that the sequence of value functions has the following (monotonicity) property $J^{\pi^0}(x) \geq J^{\pi^1}(x) \geq \cdots \geq J^*(x)\ \forall x \in X$. Thus the sequence of policies $\{\pi^k,\ k = 0, 1, \ldots\}$ generated by PI is improving. Note that the total number of stationary policies is finite whenever the action space is finite. In this particular case, we will have $J^{\pi^{k+1}}(x) = J^{\pi^k}(x)\ \forall x \in X$ for some finite k, which implies that PI obtains an optimal policy π^* in finite number of iterations. For relatively small problems (the size of the state space is less than 10^4), policy iteration is generally regarded as the fastest method for computing the optimal value function and the associated optimal policy, provided that the discount factor is sufficiently large [70].

2.2 Global Optimization

We consider the following optimization problem

$$x^* \in \arg\max_{x \in \mathcal{X}} H(x), \quad x \in \mathcal{X} \subseteq \Re^n, \tag{2.7}$$

where $\mathcal{X}$ is the solution space, and $H(\cdot) : \mathcal{X} \to \Re^+ \cup \{0\}$. We assume that the feasible region $\mathcal{X}$ is unconstrained (i.e., $\mathcal{X} = \Re^n$) or is subjected to relatively simple constraints so that the random samplings can be done easily on it; for instance, $\mathcal{X}$ is a finite set of alternatives or of the form $[a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_n, b_n]$.

In this Chapter, we review the cross-entropy (CE) method, estimation of distribution algorithms (EDAs), and the annealing adaptive search (AAS) for solving (2.7). As mentioned in Chapter 1.2, they all fall within the framework of *model-based* methods. One of the most important features of a *model-based* approach is its ability to learn and adapt during the search process. Initially, the approach starts from a global perspective, and gathers information about the "gross behavior" of the objective function by random sampling of the entire feasible region $\mathcal{X}$. As more finer details of the cost function are revealed, the searches (random sampling) are getting more and more concentrated on subregions of $\mathcal{X}$ containing high quality solutions. In a nutshell, this learning process consists of the following two steps:

1. Generating candidate solutions according to some parameterized probabilistic model.
2. Modifying the parameters of the model by using the candidate solutions in order to bias future sampling toward high quality solutions.

Thus, two crucial ingredients for any model based approaches are: (1) A probabilistic model that allows an efficient generation of candidate solutions; (2) An efficient rule for updating the parameters of the model.

2.2.1 The Cross-Entropy Method

The CE method starts with a family of parameterized probability density/mass functions $\{f(\cdot; \theta) : \theta \in \Theta\}$ over $\mathcal{X}$, where Θ is the parameter space. Instead of directly solving (2.7), the algorithm tries to solve the following estimation problem

$$\ell(\gamma) = P_\theta(H(X) \geq \gamma) = E_\theta I_{\{H(X)\geq\gamma\}},$$

where X is a random vector taking values in $\mathcal{X}$ with p.d.f./p.m.f. $f(\cdot;\theta)$, γ is some parameter, and

$$I_{\{H(x)\geq\gamma\}} = \begin{cases} 1 & \text{if } H(x) \geq \gamma, \\ 0 & \text{otherwise.} \end{cases}$$

Let us denote the maximum of (2.7) by H^*. The goal of CE is to find an optimal parameter θ^* so that the p.d.f./p.m.f. $f(\cdot,\theta^*)$ assigns maximum mass to the set of (near) optimal solutions $\{x : H(x) \geq H^*\}$. Once such a parameter is found, the resulting p.d.f./p.m.f. can be used to generate good candidate solutions to the optimization problem with high probability. However, if γ is close to H^*, then typically $\{H(X) \geq \gamma\}$ is a rare event, and estimation of the probability $\ell(\gamma)$ is a nontrivial problem. The CE method breaks down this estimation problem into a sequence of simpler estimation problems and generates a sequence of tuples $\{(\hat{\gamma}_k, \hat{\theta}_k), k = 0, 1, \ldots\}$, which converges (empirically) quickly to a small neighborhood of the optimal tuple (H^*, θ^*). The main CE optimization algorithm is summarized as follows.

Algorithm 2.2.1 (Main CE Algorithm for Optimization) *Let $\rho \in (0,1)$ be the fraction of the best samples that will be used in parameter updating, and N be the number of samples at each iteration.*

1. *Choose the initial parameter $\hat{\theta}_0 \in \Theta$. Set the iteration counter $k = 0$.*

2. *Draw random samples $X_k^1, \ldots, X_k^N$ according to $f(\cdot, \hat{\theta}_k)$. Calculate the sample $(1-\rho)$-quantile by ordering $H(X_N^i)$ $i = 1, \ldots, N$ from the smallest to largest and then setting*

$\hat{\gamma}_k := H_{(\lceil \rho N \rceil)}$, where $H_{(i)}$ is the ith order statistic of the ordered sample performance and $\lceil \rho N \rceil$ indicates the integer part of ρN.

3. *Calculate the new parameter $\hat{\theta}_{k+1}$ by solving the optimization problem*

$$\hat{\theta}_{k+1} := \arg\max_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} I_{\{H(X_k^i) \geq \hat{\gamma}_k\}} \ln f(X_k^i, \theta).$$

4. *If for some $k \geq d$, say $d = 5$,*

$$\hat{\gamma}_k = \hat{\gamma}_{k-1} = \cdots = \hat{\gamma}_{k-d},$$

then terminate; otherwise set $k = k+1$ and reiterate from Step 2.

The deterministic version of Algorithm 2.2.1 is also presented below.

Algorithm 2.2.2 (Deterministic Version of the CE Method)

1. *Choose the initial parameter $\theta_0 \in \Theta$. Set $k = 0$.*

2. *Calculate the $(1-\rho)$-quantile γ_k as*

$$\gamma_k := \max \left\{ l : P_{\theta_k}(H(X) \geq l) \geq \rho \right\}.$$

3. *Compute the new parameter by solving the following problem*

$$\theta_{k+1} := \arg\max_{\theta \in \Theta} E_{\theta_k} \left[I_{\{H(X) \geq \gamma_k\}} \ln f(X, \theta) \right].$$

4. *If for some $k \geq d$, say $d = 5$,*

$$\gamma_k = \gamma_{k-1} = \cdots = \gamma_{k-d},$$

then terminate; otherwise set $k = k+1$ and reiterate from Step 2.

2.2.2 The Estimation of Distribution Algorithms

The EDAs were first introduced in the field of evolutionary computation. However, unlike evolutionary algorithms, they do not rely on the "genetic" principle anymore (e.g., the crossover and mutation mechanisms in classical evolutionary algorithms); instead, in each iteration, they build an explicit probabilistic model (probability distribution) of promising solutions in the search space. New candidate solutions are created by sampling from this distribution. In a general level, an EDA can be concisely described as follows.

Algorithm 2.2.3 (Estimation of Distribution Algorithm) *Let N be the size of the population at each iteration.*

1. *Generate the initial population D_0 (N candidate solutions) randomly (e.g., uniformly) from the solution space. Set the iteration counter $k = 0$.*
2. *Construct a set of promising solutions D_k^S by selecting $S \leq N$ candidate solution from D_k according to a selection scheme.*
3. *Estimate $P_k(x) := P(x|D_k^S)$ for all $x \in \mathcal{X}$, i.e., the probability distribution of solution x being among the selected solutions D_k^S.*
4. *Construct a new population D_{k+1} by sampling N candidate solutions from $P_k(x)$.*
5. *If a stopping criterion is met, then terminate; otherwise set $k = k + 1$ and reiterate from step 2.*

The performance of a particular EDA are mainly determined by the construction and estimation of the probabilistic model $P_k(\cdot)$. More accurate models ensure better performance of the algorithm, however they are often more complicated and expensive to build. In combinatorial domains, if the random vector $X \in \mathcal{X}$ consists of n discrete variables, i.e., $X = (X_1, X_2, \ldots, X_n)$, and each variable X_i can take on m values, then a complete description of the joint probability distribution of X requires $m^n - 1$ parameters, and to estimate all these parameters is clearly impractical. In practice, in order to reduce the number of parameters used to represent the joint distribution, simplifying assumptions are made about the structure of the distribution. For instance, consider the case where $n = 3$ and $m = 3$. A precise description of the joint distribution requires 26 parameters: 2 for the distribution of X_3, 6 for the conditional distribution $P(X_2 = y|X_3 = z)$, and 18 for $P(X_1 = x|X_2 = y, X_3 = z)$. If we assume that given X_2, X_1 is independent of X_3, then only 14 parameters are required. Finally, if all variables are assumed to be independent, then the joint distribution of X is determined by the univariate marginal distribution of X_1, X_2, and X_3, which in turn requires only 6 parameters. Thus, as we can see, there is often a tradeoff between accuracy and efficiency. When categorized by the complexity of the underlying probabilistic models employed, there are a number of different particular instantiations of EDAs, ranging from the simple *Univariate Marginal Density Algorithm* (UMDA) [59], where all components of an individual solution are assumed to be independent, to *Bayesian Optimization Algorithm* (BOA) which uses Bayesian nets as the probabilistic model. Please refer to [53] and [60] for a review.

2.2.3 Annealing Adaptive Search

The annealing adaptive search method was originally developed to understand the behavior of the classical simulated annealing algorithm. The method, in its idealized form, assumes that the samples can be generated exactly from a sequence of Boltzmann distributions (in a maximization context) given by

$$g_{T_k}(x) = \frac{e^{H(x)/T_k}}{\int_{\mathcal{X}} e^{H(x)/T_k} \nu(dx)},$$

where ν is the Lebesgue or discrete measure on the solution space, and T_k is the temperature parameter at the kth iteration, which is usually taken to be a function (cooling schedule) of the past sample/candidate solution visited. The idealized version of AAS, taken from [89], is presented below.

Algorithm 2.2.4 *1. Generate a solution X_0 uniformly from the solution space $\mathcal{X}$. Set $k = 0$, $Y_0 = H(X_0)$, $Y_* = Y_0$, $X_* = X_0$, and $T_0 = \tau(X_*)$, where $\tau(\cdot)$ is a positive real-valued nondecreasing cooling schedule.*

2. Generate X_{k+1} from the Boltzmann distribution with temperature parameter T_k.

3. If $H(X_{k+1}) > Y_k$, set $Y_{k+1} = H(X_{k+1})$, $Y_* = Y_{k+1}$, $X_* = X_{k+1}$. Set $T_{k+1} = \tau(Y_*)$. Otherwise, set $Y_{k+1} = Y_k$ and $T_{k+1} = T_k$.

4. Set $k = k + 1$ and return to Step 2 until some specified stopping rule is satisfied.

AAS has some attractive theoretical properties. For example, it is shown in [74] that for a particular cooling schedule of the temperature parameter, the expected number of improving samples/solutions (in terms of their performance) and the number of function evaluations both grow only linearly with the problem dimension. However, as noted earlier, in order to implement the method in practice, AAS needs to be used in conjunction with various efficient sampling techniques. This is an active area that has received much attention both in the past and present. Since the technical details is beyond the scope of this research, we refer interested readers to the work of [74] and [89].

Chapter 3

An Adaptive Multi-stage Sampling Algorithm for Solving Finite Horizon Markov Decision Processes

In this chapter, we propose a simulation-based framework for approximately solving general finite horizon MDPs with large state spaces. For a given MDP with horizon T, the method can be interpreted as an efficient search method for a decision tree with depth T, where each node of the tree represents a state, with the root node corresponding to an initial state, and each edge of the tree signifies a sampling of a given action. The method employs a depth first search for generating sample paths from the initial state to the final state (i.e., when the finite horizon T is reached) and uses backtracking to estimate the value functions at previously visited states, where the estimated value function of a certain node/state is taken to be the weighted average of the Q-values at the successive child nodes/states. We show that the estimated value function at the initial state produced by the algorithm not only converge to the true optimal value but also does so in an "efficient" way, with the worst-case bias bounded by a quantity that converges to zero at rate of $O\left(\sum_{t=0}^{T-1} \frac{\ln N_t}{N_t}\right)$, where N_t is the total number of samples that are used per state sampled in stage t. Given that the action space size is $|A|$, the worst-case running time-complexity of the algorithm is $O\left((|A| \max_{t=1,\dots,T} N_t)^T\right)$, which is independent of the state space size but is dependent on the size of the action space due to the requirement that each action be sampled at least once at each sampled state.

A similar sampling strategy (i.e., the recursive tree sampling structure) was previously used in [49] to create an on-line, near-optimal planning algorithm for solving large

MDPs. However, their approach differs from ours in the way actions are sampled. Their method employs a straightforward nonadaptive sampling scheme, where each action is always sampled for a prespecified fixed number of times. Obviously this scheme is generally sub-optimal, which could often lead to a waste of computational resources, especially when the computational budget is tight. Our method, in contrast, adaptively chooses which action to sample as the sampling process proceeds, and concentrate most of the sampling on the action with high variability, which could yield the most computational benefits in cases where the sampling cost is relatively expensive.

The adaptive sampling idea in our approach originates from the expected *regret* analysis of the multi-armed bandit problem developed by [52]. In particular, we exploit the recent finite-time analysis work by [8] that elaborated [1]. The objective of these problems is to play as often as possible the machine that yields the highest (expected) reward. The optimal strategy (policy) must balance between playing the machine that is empirically best thus far (exploitation), i.e., the machine has the highest sample mean, and trying to find a better machine (exploration) that actually has a higher expectation but might have a lower sample mean thus far due to statistical variation. The expected loss due to not always playing the true optimal machine is called regret, which quantifies the exploration/exploitation dilemma in the search for the true (unknown in advance) "optimal" machine. Lai and Robbins [52] showed that for an optimal strategy the regret grows at least logarithmically in the number of machine plays, and recently Auer et al. [8] showed that the logarithmic regret is also achievable uniformly over time with a simple and efficient sampling algorithm for arbitrary reward distributions with bounded support. We incorporate their results into a sampling-based process for finding an optimal action in a state for a single stage of a finite horizon MDP by appropriately converting the definition of regret into the difference between the true optimal value and the approximate value yielded by the sampling process. We then extend the one-stage sampling process into multiple stages in a recursive manner, leading to a multi-stage (sampling-based) approximation algorithm for solving MDPs.

3.1 Related Work

The multi-armed bandit problems have been studied extensively for many years, however, the literature applying the theory of the multi-armed bandit problem to derive a probably convergent framework for solving general MDPs is very few. The closest related work is probably that of Agrawal et al. [2], who considered a controlled finite-state/action-space Markov chain problem with infinite horizon average reward criterion. In their setting, transition probabilities and initial distribution are parameterized by an unknown parameter θ selected from some known finite parameter, with each fixed parameter θ leading to an ergodic Markov chain. They assume that for each θ, there exists a unique optimal stationary policy. They consider a finite-horizon loss function defined over all θ's based on the regret of [52], and regard the optimal stationary policy for the average reward as an approximation for an optimal nonstationary policy that minimizes the loss for the finite horizon. By then using the optimal stationary policy for the average reward for each θ, they develop an adaptive but rather complex policy, the performance of which is bounded in terms of the horizon size of the loss function, which vanishes as the size increases. The adaptiveness comes from the use of the multi-armed bandit theory for the stationary control laws. In other words, the arm corresponds to a particular stationary law or policy, but not a particular action in the action space.

3.2 Adaptive Sampling Algorithm

3.2.1 Background

We consider the MDP problem $M = (X, A, \{P_t, t = 0, 1, \ldots\}, \{R_t, t = 0, 1, \ldots\}, \alpha)$ with finite horizon length T, finite state space X, finite action space with $|A| > 1$, and bounded non-negative one-stage cost function R_t. Again for simplicity (and without loss of generality), we assume that every action is admissible in every state.

At stage $t < T$, for a given state x, we define the optimal discounted reward-to-go at state x from stage t as

$$J_t^*(x) = \sup_{\boldsymbol{\pi} \in \Pi} E\left[\sum_{i=t}^{T-1} \alpha^i R_i(x_i, \pi_i(x_i)) \Big| x_t = x \right], x \in X, 0 < \alpha \leq 1, t = 0, ..., T-1, \tag{3.1}$$

with $J_T^*(x) = 0$ for all $x \in X$, where Π is the set of all possible nonstationary Markovian policies $\boldsymbol{\pi} = \{\pi_t | \pi_t : X \rightarrow A, t \geq 0\}$, and the assumption that we have the zero terminal reward function (for simplicity) can be relaxed with an arbitrary terminal reward function. Our goal is to estimate for a given initial state x, the optimal discounted total reward (thereby obtaining an approximate optimal policy) $J_0^*(x)$. As mentioned earlier, the objective of multiarmed bandit problems is to identify the machine that have the highest reward. Therefore, for ease of exposition, it is natural for us to consider in (3.1) a slightly different version of the MDP model introduced in Chapter 2, i.e., maximizing the reward instead of minimizing the cost. However, we remark that all results can be easily extended to a minimization context, we will come back to this issue later in Chapter 3.4.

By Theorem 2.1.1, the optimal reward-to-go J_t^* can be obtained recursively as follows: for all $x \in X$ and $t = 0, ..., T-1$,

$$\begin{aligned} J_t^*(x) &= \max_{a \in A}(Q_t^*(x,a)), \text{ where we define} \\ Q_t^*(x,a) &= R_t(x,a) + \alpha \sum_{y \in X} P_{x,y|t}(a) J_{t+1}^*(y). \end{aligned} \tag{3.2}$$

The right hand side of (3.2) is basically the sum of one-stage cost plus the expected value of the future optimal cost-to-go, therefore a natural way to estimate $Q_t^*(x,a)$ is to use its sample average approximation $\hat{Q}_t(x,a)$ given by

$$\hat{Q}_t(x,a) = R_t(x,a) + \alpha \frac{1}{N_{a,t}^x} \sum_{y \in S_a^x} \hat{J}_{t+1}^{N_{t+1}}(y), \tag{3.3}$$

where S_a^x is the *multiset* (in which the same element may appear for more than once) of independently sampled next states according to the transition probability $P_{x,\cdot|t}(a)$, and $N_{a,t}^x := |S_a^x| \geq 1$ is the cardinality of the set S_a^x with $\sum_{a \in A} N_{a,t}^x = N_t$ for a fixed $N_t \geq |A|$ for all $x \in X$, and $\hat{J}_{t+1}^{N_{t+1}}(y)$ is an estimate of the optimal cost-to-go at the sampled next state y. Note that the number of next state samples depends on the state x, action a, and stage t. If we further estimate the optimal value $J_t^*(x)$ by a weighted sum

$$\hat{J}_t^{N_t}(x) = \sum_{a \in A} \frac{N_{a,t}^x}{N_t} \hat{Q}_t(x,a).$$

Then we have the following recursive relationship

$$\hat{J}_t^{N_t}(x) := \sum_{a \in A} \frac{N_{a,t}^x}{N_t} \left(R_t(x,a) + \alpha \frac{1}{N_{a,t}^x} \sum_{y \in S_a^x} \hat{J}_{t+1}^{N_{t+1}}(y) \right), i = 0, ..., T-1,$$

with $\hat{J}_T^{N_T}(x) = J_T^*(x) = 0$ for all $x \in X$.

In the above definition, the total number of sampled (next) states is $O(N^T)$ with $N = \max_{t=0,...,T-1} N_t$, which is independent of the state space size. To carry out the above recursion, we need to determine the value of $N_{a,t}^x$ for $t = 0, ..., T-1$, $a \in A$, and $x \in X$. An obvious way is to use the straightforward non-adaptive approach, and use the same fixed value of $N_{a,t}^x$ for all x, a, and t. But here we consider an adaptive allocation rule (sample scheme), in particular, we want to adaptively choose the value of $N_{a,t}^x$ in such a way that the expected difference between $\hat{J}_0^{N_0}(x)$ and $J_0^*(x)$ is bounded as a function of $N_{a,t}^x$ and N_t, $t = 0, \ldots, T-1$, and the bound goes to zero as N_t goes to infinity.

The main idea behind the adaptive allocation rule is based on a simple interpretation of the regret analysis of the multi-armed bandit problem, where plays of the ith machine ($1 \leq i \leq m$, m is the total number of machines) yield i.i.d. random rewards with unknown mean μ_i, and the goal is to play as often as possible the machine corresponding to the maximum mean μ^*. The rewards across different machines are also assumed to be independently generated. Let $C_i(n)$ be the number of times the ith machine has been played by an algorithm during the first n plays. We define the *expected regret* $\rho(n)$ of an algorithm after n plays by

$$\rho(n) = \mu^* n - \sum_{i=1}^{m} \mu_i E[C_i(n)].$$

Lai and Robbins [52] characterized an "optimal" algorithm such that the best machine, which is associated with μ^*, is played exponentially more often than any other machine, at least asymptotically. That is, they showed that playing machines according to an (asymptotically) optimal algorithm leads to $\rho(n) = \Theta(\ln n)$ as $n \to \infty$ under mild assumptions on the reward distributions. However, obtaining an optimal algorithm (proposed by Lai and Robbins) is often very difficult, so Agrawal [1] derived a set of simple algorithms that achieve the asymptotic logarithmic regret behavior, using a form of *upper confidence bounds*. The use of the upper confidence bound leads us to trade-off between exploitation and exploration, giving a criterion of which of the two between exploitation and exploration to be selected. For example, let $\bar{n}$ be the number of overall plays (for all machines) so far, and let $\hat{\mu}_i(\bar{n})$ be the sample mean reward accumulated by playing machine i. During the plays, we are tempted to take the machine with the maximum current sample mean (exploitation). However, $\hat{\mu}_i(\bar{n})$ is just an estimate of the true mean, which may contain high variability. Therefore, always playing the machine that yields the best current sample mean is obviously non-optimal, it is also desirable to play other machines occasionally (exploration). To account for the variability in the estimation, we try to find a function $\sigma_i(\bar{n})$ such that the true mean μ_i falls in the confidence interval $(\hat{\mu}_i(\bar{n}) - \sigma_i(\bar{n}), \hat{\mu}_i(\bar{n}) + \sigma_i(\bar{n}))$ with high probability. Agrawal's algorithm is to choose the machine with the highest upper confidence bound at each play over time. For bounded rewards, [8] propose simple upper confidence-bound based algorithms that achieve the logarithmic regret uniformly over time, rather than only asymptotically, and our sampling algorithm primarily builds on their results.

To see how we incorporate the confidence bound idea into an adaptive allocation rule for finite horizon MDPs, we consider first only the one-stage problem (i.e., $T = 1$). For this problem, by definition we know the value of $J_1^*(x)$ for all $x \in X$, and our goal is to estimate $J_0^*(x)$. From (3.2), it is obvious we need to obtain a viable estimate for $Q_0^*(x, a^*)$, where $a^* \in \arg\max_{a \in A}(Q_0^*(x, a))$. The search for a^* corresponds to the search for the best machine in the multi-armed bandit problem. We start by sampling each possible action once at x, which leads to the next state according to $P_{x,\cdot|0}(a)$ and reward $R_0(x, a)$. The next action to sample is the one that achieves the maximum among the current estimates of $Q_0^*(x, a)$ plus its current upper confidence bound (see (3.5)), where the estimate $\hat{Q}_0(x, a)$ is given by the immediate reward plus the *sample mean* of J_1^*-values at the sampled next states that have been sampled so far (see (3.6)). The above procedure is repeated until a prespecified total number of sampling budget is consumed, see, in particular, the **Loop** step in Figure 3.1.

Given the total number of samples N_0 for state x at the initial stage, $N^x_{a,0}$ denotes the number of times action a has been sampled. If the sampling is done appropriately, we might expect that in the long run, the optimal actions will be sampled significantly more often than other non-optimal actions, thus $N^x_{a,0}/N_0$ provides a good estimate of the likelihood whether action a is optimal in state x. As a result, in the limit as $N_0 \to \infty$, we should have $\lim_{N_0\to\infty} \sum_{a\in A^*} N^x_{a,0}/N_0 \to 1$, where A^* denotes the set of all optimal actions. For this reason, we use a weighted (by $N^x_{a,0}/N_0$) sum of the currently estimated value of $Q^*_0(x,a)$ over A to approximate $J^*_0(x)$ (see (3.7)). Therefore, as the weighted sum concentrates on a^* as the sampling proceeds, we will have the convergence of the estimate $\hat{J}^{N_0}_0(x)$ to $J^*_0(x)$.

Remark 3.2.1 *Throughout this Chapter, the notation O used in the sense that for given two functions f and g, $f(n) = O(g(n))$ if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = c$ for some constant $c > 0$, and the notation Θ is used in that there exist positive constants c_1, c_2, and n_0 such that $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$ for all $n \ge n_0$ ([25]). The O and Θ-notations are often called asymptotic upper bound and asymptotically tight bound, respectively, for the asymptotic running time of an algorithm.*

3.2.2 Algorithm description

The adaptive multi-stage sampling (AMS) algorithm is essentially a recursive extension of the one-stage sampling approach describe in preceding two paragraphs. The basic algorithmic procedure is given in Figure 3.1. The inputs to the algorithm are a state $x \in X$, the total number of samples $N_t \ge |A|$ allowed at stage t, and the output of algorithm is an estimate of the true optimal reward-to-go from state x $\hat{J}^{N_t}_t(x)$. The AMS algorithm itself is recursively called to estimate $J^*_k(y)$ whenever we need to calculate the value $\hat{J}^{N_k}_k(x)$ for

Adaptive Multi-stage Sampling (AMS)

- **Input:** a state $x \in X$, $N_t \geq |A|$, and stage t. **Output:** $\hat{J}_t^{N_t}(x)$.
- **Initialization:** Sample each action $a \in A$ sequentially once at state x and set

$$\hat{Q}_t(x,a) = \begin{cases} 0 \text{ if } t = T \text{ go to } \textbf{Exit} \\ R_t(x,a) + \alpha \hat{J}_{t+1}^{N_{t+1}}(y) \text{ if } t \neq T, \end{cases} \tag{3.4}$$

  where y is the sampled next state according to $P_{x,\cdot|t}(a)$, set the total current number of samples $\bar{n} = |A|$.

- **Loop:** Sample the action $\hat{a}^*$ (estimate of the true a^*) that has the best upper confidence bound

$$\max_{a \in A} \left(\hat{Q}_t(x,a) + \sqrt{\frac{2 \ln \bar{n}}{N_{a,t}^x}} \right), \tag{3.5}$$

  where $N_{a,t}^x$ denotes the number of times action a has been sampled, and $\hat{Q}_t$ is defined by

$$\hat{Q}_t(x,a) = R_t(x,a) + \alpha \frac{1}{N_{a,t}^x} \sum_{y \in S_a^x} \hat{J}_{t+1}^{N_{t+1}}(y), \tag{3.6}$$

  where S_a^x is the set of sampled next states so far with $|S_a^x| = N_{a,t}^x$ with respect to $P_{x,\cdot|t}(a)$.

  - Update $N_{\hat{a}^*,t}^x \leftarrow N_{\hat{a}^*,t}^x + 1$ and $S_{\hat{a}^*}^x \leftarrow S_{\hat{a}^*}^x \cup \{y'\}$, where y' is the newly sampled next state by $\hat{a}^*$.
  - Update $\hat{Q}_i(x, \hat{a}^*)$ with the $\hat{J}_{t+1}^{N_{t+1}}(y')$ value.
  - $\bar{n} \leftarrow \bar{n} + 1$. If $\bar{n} = N_t$, then exit **Loop**.

- **Exit:** Set $\hat{J}_t^{N_t}(x)$ such that

$$\hat{J}_t^{N_t}(x) = \begin{cases} \sum_{a \in A} \frac{N_{a,t}^x}{N_t} \hat{Q}_t(x,a) \text{ if } t = 0, ..., T-1 \\ 0 \text{ if } t = T. \end{cases} \tag{3.7}$$

  and return $\hat{J}_t^{N_t}(x)$.

Figure 3.1: Adaptive multi-stage sampling algorithm (AMS) description

a state $y \in X$ at stage k in the **Initialization** and **Loop** subroutines of the algorithm. In particular, we need to call AMS recursively (at Equation (3.4) and (3.6)). The initial call

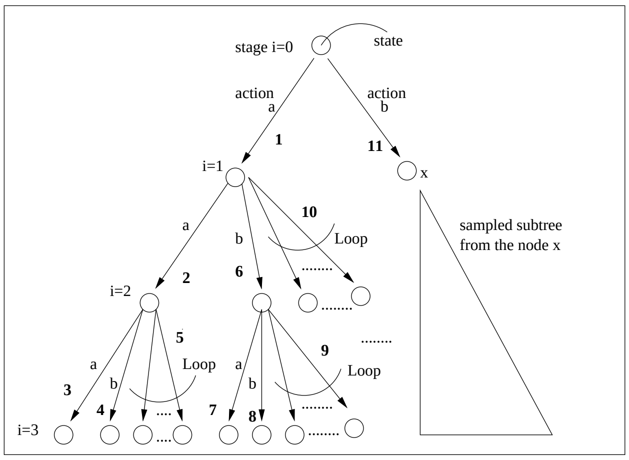


Figure 3.2: The sequence of the recursive calls made in **Initialization** of the AMS algorithm. Each node corresponds to a state and each arrow with noted action signifies a sampling (and a recursive call). The bold-face number near each arrow is the sequence number for the recursive calls made. For simplicity, the entire **Loop** process is signified by one call number.

to AMS is done with $t = 0$ and the initial state x_0, and every sampling is done independently of the previously done samplings. Figure 3.2 graphically illustrates the sequence of calls with two actions and $T = 3$ for the **Initialization** portion. We remark that this sampling strategy, as depicted in Figure 3.2, resembles the recursive decision tree in the same spirit as [49] use for planning algorithms, and the non-recursive simulated/sampling trees [17] use for an American-style option pricing problem and [33] use in a more general MDP setting. However, as mentioned before, all those works use non-adaptive sampling, in the sense that the number of samples for each action is pre-specified.

Now let M_t be the number of recursive calls made to compute $\hat{J}_t^{N_t}$ in the *worst* case. At stage t, AMS makes at most $M_t = |A|N_t M_{t+1}$ recursive calls (in **Initialization** and **Loop**). Thus, the worst case running time complexity of AMS is $M_0 = O((|A| \max_t N_t)^T)$. In contrast, backward induction has $O(T|A||X|^2)$ running time complexity (see, e.g., [16]).

Therefore, the main benefit of AMS is independence from the state space size, but this comes at the expense of exponential (versus linear, for backwards induction) dependence on both the action space and the horizon length.

3.3 Convergence Analysis

In this Chapter, we study the convergence properties of the AMS algorithm. In particular, we show that the final estimate of the optimal value function produced by the algorithm is asymptotically unbiased, and the worst possible bias is uniformly bounded by a quantity that converges to zero at rate $O\left(\sum_{t=0}^{T-1} \frac{\ln N_t}{N_t}\right)$.

We first consider a special case of the AMS algorithm, a non-recursive one-stage sampling algorithm (OSA) results from by applying AMS to the one-stage approximation problem described earlier in Chapter 3.2.1. The algorithm is illustrated in Figure 3.3 with a *stochastic value function* U defined over X. $U(x)$ for $x \in X$ is a *nonnegative bounded random variable* with *unknown* distribution for all $x \in X$. $U(x)$ can be viewed as the outcome/observation of a black box corresponding to a input x, where as before, when x is given to the black box, we assume that the observations at different time instances are independent of each other, and are identically distributed according to the unknown distribution. Let

$$U_{\max} = \max_{x,a} \left(R(x,a) + \alpha \sum_{y \in X} P_{x,y}(a) E[U(y)] \right),$$

and assume for the moment that $U_{\max} \leq 1$. Note that since we are considering the one-stage problem, we have dropped the dependencies on stage t in both R and P. However, we should keep in mind that this setting, as well as all subsequent results, hold for every stage $t = 0, \ldots, T-1$.

We now interpret the OSA in the context of a $|A|$-armed bandit problem, where each

One-stage Sampling Algorithm (OSA)

- **Input:** a state $x \in X$ and $n \geq |A|$.
- **Initialization:** Sample each action $a \in A$ once at state x and set

$$\tilde{Q}(x,a) = R(x,a) + \alpha U(y),$$

  where $y \sim P_{x,\cdot}(a)$ is the sampled next state. Set $\bar{n} = |A|$.
- **Loop:** Sample an action a^* that achieves

$$\max_{a \in A} \left(\tilde{Q}(x,a) + \sqrt{\frac{2 \ln \bar{n}}{T_a^x(\bar{n})}} \right),$$

  where $T_a^x(\bar{n})$ is the number of times action a has been sampled so far at state x, $\bar{n}$ is the overall number of samples done so far, and

$$\tilde{Q}(x,a) = R(x,a) + \alpha \frac{1}{T_a^x(\bar{n})} \sum_{y \in \Lambda_a^x(\bar{n})} U(y),$$

  where $\Lambda_a^x(\bar{n})$ is the set of sampled next states so far with $|\Lambda_a^x(\bar{n})| = T_a^x(\bar{n})$.
  - Update $T_{a^*}^x(\bar{n}) \leftarrow T_{a^*}^x(\bar{n}) + 1$ and $\Lambda_{a^*}^x(\bar{n}) \leftarrow \Lambda_{a^*}^x(\bar{n}) \cup \{y'\}$, where y' is the newly sampled next state by a^*.
  - Update $\tilde{Q}(x,a^*)$ with $U(y')$.
  - $\bar{n} \leftarrow \bar{n} + 1$. If $\bar{n} = n$, then exit **Loop**.
- **Exit:** Set $\tilde{J}^n$ such that

$$\tilde{J}^n(x) = \sum_{a \in A} \frac{T_a^x(n)}{n} \tilde{Q}(x,a). \tag{3.8}$$

Figure 3.3: One-stage sampling algorithm (OSA) description

action a corresponds to a gambling machine. Successive plays of machine a yield "bandit rewards" which are independent and identically distributed according to an unknown distribution δ_a with unknown expectation

$$Q(x,a) = R(x,a) + \alpha \sum_{y \in X} P_{x,y}(a) E[U(y)],$$

and are independent across machines or actions.

In OSA, $T_a^x(n)$ represents the number of times machine a has been played (or action a has been sampled) during the n plays. Define the *expected regret* $\rho(n)$ of OSA after n plays by

$$\rho(n) = J(x)n - \sum_{a=1}^{|A|} Q(x,a)E[T_a^x(n)], \text{ where } J(x) = \max_{a \in A} Q(x,a).$$

We now state a key theorem, which will be the basis of our convergence results for the OSA algorithm, whose proof is given in [8].

Theorem 3.3.1 *For all $|A| > 1$, if OSA is run on $|A|$-machines having arbitrary bandit reward distribution $\delta_1, ..., \delta_{|A|}$ with $U_{\max} \leq 1$, then*

$$\rho(n) \leq \sum_{a:Q(x,a)<J(x)} \left[\frac{8 \ln n}{J(x) - Q(x,a)} + (1 + \frac{\pi^2}{3})(J(x) - Q(x,a)) \right],$$

where $Q(x,a)$ is the expected value of bandit rewards with respect to δ_a.

The convergence of the OSA algorithm is summarized in the following lemma.

Lemma 3.3.1 *With the stochastic value U defined earlier with $U_{\max} \leq 1$, and suppose the total number of sample allowed by OSA is n. Then we have, for all $x \in X$,*

$$E[\tilde{J}^n(x)] \to J(x) \text{ as } n \to \infty,$$

where $J(x)$ is the true optimal value function at state x for the one-stage problem, i.e.,

$$J(x) = \max_{a \in A} \left(R(x,a) + \alpha \sum_{y \in X} P_{x,y}(a) E[U(y)] \right).$$

Proof: Note that $\max_a (J(x) - Q(x,a)) \leq U_{\max}$. We define the set of nonoptimal actions for x as $\phi(x) = \{a | Q(x,a) < J(x), a \in A\}$. Define $\beta(x)$ for $\phi(x) \neq \emptyset$ such that

$$\beta(x) = \min_{a \in \phi(x)} (J(x) - Q(x,a)) \tag{3.9}$$

and note that $0 < \beta(x) \leq U_{\max}$. Let

$$\tilde{J}(x) = \sum_{a=1}^{|A|} \frac{T_a^x(n)}{n} Q(x,a).$$

We have by Theorem 3.3.1,

$$\begin{aligned} 0 \leq J(x) - E[\tilde{J}(x)] &= \frac{\rho(n)}{n} \leq \frac{8(|A|-1)\ln n}{n\beta(x)} + (1+\frac{\pi^2}{3}) \cdot \frac{(|A|-1)U_{\max}}{n} \\ &\leq \frac{C_1 \ln n}{n} + \frac{C_2}{n}, \end{aligned} \tag{3.10}$$

where C_1 and C_2 are some constants. Since X is finite, there exists a constant $C > 0$ such that $0 < C \leq \min_{x \in X} \beta(x)$ and also that $\rho(n) = 0$ if $\phi(x) = \emptyset$. By the definition of $\tilde{J}^n(x)$, (cf. (3.8)), it follows that

$$\begin{aligned} J(x) - E[\tilde{J}^n(x)] &= J(x) - E[\tilde{J}(x) - \tilde{J}(x) + \tilde{J}^n(x)] \\ &= J(x) - E[\tilde{J}(x)] + E\left[\sum_{a \in A} \frac{T_a^x(n)}{n} \Big(Q(x,a) - \tilde{Q}(x,a)\Big)\right]. \end{aligned} \tag{3.11}$$

Clearly by (3.10), the first term $J(x) - E[\tilde{J}(x)]$ above is bounded by zero from below with convergence rate of $O(\frac{\ln n}{n})$. We now show that the last term in (3.11) is zero.

Let $Y_j \sim \{P_{x,\cdot}(a)\}$ denote the (i.i.d.) jth next state sampled from the same starting state x with same action a. Then, $T_a^x(n)$ for every finite n is a *stopping time* (cf. e.g., [64], p.104) for $\{Y_j\}$, since $T_a^x(n) \leq n < \infty$ and the event $\{T_a^x(n) = k\}$ is independent of $\{Y_{k+1}, \ldots\}$. It follows that

$$\begin{aligned} &E\Big[\sum_{a \in A} \frac{T_a^x(n)}{n} \Big(Q(x,a) - \tilde{Q}(x,a)\Big)\Big] \\ &= E\Big[\sum_{a \in A} \frac{T_a^x(n)}{n} \Big(R(x,a) + \alpha E[U(Y_j)] - R(x,a) - \alpha \frac{1}{T_a^x(n)} \sum_{j=1}^{T_a^x(n)} U(Y_j)\Big)\Big] \\ &= \frac{\alpha}{n} \Big(\sum_{a \in A} E[T_a^x(n)] E[U(Y_j)] - \sum_{a \in A} E\Big[\sum_{j=1}^{T_a^x(n)} U(Y_j)\Big]\Big) = 0, \end{aligned}$$

by applying Wald's equation.

Therefore, the convergence follows directly from (3.10) and (3.11). ∎

We are now ready to state the main convergence theorem for the AMS algorithm, whose proof is based upon a straightforward inductive application of Lemma 3.3.1.

Theorem 3.3.2 *Assume that the one-stage reward function is uniformly bounded by $\frac{1}{T}$, i.e., $R_{\max} = \max_{x,a,t} R_t(x,a) \leq \frac{1}{T}$. Suppose AMS is run with a given (arbitrary) initial state x and input N_t, $t = 0, ..., T-1$. Then*

(1)

$$\lim_{N_t \to \infty,\ \forall\ t=0,\ldots,T} E[\hat{J}_0^{N_0}(x)] = J_0^*(x).$$

Moreover, the worst possible bias induced by the algorithm is bounded by a quantity that converges to zero at rate $O\big(\sum_{t=0}^{T-1} \frac{\ln N_t}{N_t}\big)$, i.e.,

(2)

$$J_0^*(x) - E[\hat{J}_0^{N_0}(x)] \leq O\left(\sum_{t=0}^{T-1} \frac{\ln N_t}{N_t}\right), x \in X,$$

Proof: At stage $T-1$, by the definition of $\hat{J}_{T-1}^{N_{T-1}}$,

$$\begin{aligned} \hat{J}_{T-1}^{N_{T-1}}(x) &= \sum_{a \in A} \frac{N_{a,T-1}^x}{N_{T-1}} \left(R_{T-1}(x,a) + \alpha \frac{1}{N_{a,T-1}^x} \sum_{y \in S_a^x} \hat{J}_T^{N_T}(y) \right) \\ &\leq \sum_{a \in A} \frac{N_{a,T-1}^x}{N_{T-1}} \left(R_{\max} + \alpha \cdot 0 \right) = R_{\max}, x \in X. \end{aligned}$$

It follows that at stage $T-2$

$$\begin{aligned} \hat{J}_{T-2}^{N_{T-2}}(x) &= \sum_{a \in A} \frac{N_{a,T-2}^x}{N_{T-2}} \left(R_{T-2}(x,a) + \alpha \frac{1}{N_{a,T-2}^x} \sum_{y \in S_a^x} \hat{J}_{T-1}^{N_{T-1}}(y) \right) \\ &\leq \sum_{a \in A} \frac{N_{a,T-2}^x}{N_{T-2}} \left(R_{\max} + \alpha R_{\max} \right) = R_{\max}(1+\alpha), x \in X. \end{aligned}$$

And by induction, we have for all $x \in X$ and $t = 0, ..., T-1$,

$$\hat{J}_t^{N_t}(x) \leq R_{\max} \sum_{i=0}^{T-t-1} \alpha^i \leq R_{\max}(T-t) \leq 1,$$

where the last inequality follows from the assumption $R_{\max} T \leq 1$.

Therefore, from Lemma 3.3.1 with $U_{\max} = R_{\max}(T-t) \leq 1$, we have for $t = 0, ..., T-1$, and for arbitrary $x \in X$,

$$E[\hat{J}_t^{N_t}(x)] \overset{N_t \to \infty}{\longrightarrow} \max_{a \in A} \left(R_t(x,a) + \alpha \sum_{y \in X} P_{x,y|t}(a) E[\hat{J}_{t+1}^{N_{t+1}}(y)] \right).$$

But for arbitrary $x \in X$, because $\hat{J}_T^{N_T}(x) = J_T^*(x) = 0, x \in X$,

$$E[\hat{J}_{T-1}^{N_{T-1}}(x)] \overset{N_{T-1} \to \infty}{\longrightarrow} J_{T-1}^*(x),$$

which in turn leads to $E[\hat{J}_{T-2}^{N_{T-2}}(x)] \to J_{T-2}^*(x)$ as $N_{T-2} \to \infty$ for arbitrary $x \in X$, and by an inductive argument, we have that

$$\lim_{N_t \to \infty \ \forall \ t=0,\ldots,T-1} E[\hat{J}_0^{N_0}(x)] = J_0^*(x) \text{ for all } x \in X,$$

which completes the first part of the proof.

To show the second part, we define the space of bounded real-valued measurable functions on X by $B(X)$, and at stage t, we also define an operator $\mathcal{T}_t : B(X) \to B(X)$ as

$$\mathcal{T}_t(\Phi)(x) = \max_{a \in A} \left\{ R_t(x,a) + \alpha \sum_{y \in X} P_{x,y|t}(a) \Phi(y) \right\}, \ \Phi \in B(X), \ x \in X, \ t = 0, \ldots, T-1. \tag{3.12}$$

In the proof of Lemma 3.3.1 (see (3.11)), we showed that for $t = 0, ..., T-1$,

$$\mathcal{T}_t(E[\hat{J}_{t+1}^{N_{t+1}}(x)]) - E[\hat{J}_t^{N_t}(x)] \leq O\left(\frac{\ln N_t}{N_t}\right), x \in X.$$

Therefore, we have

$$\mathcal{T}_0(E[\hat{J}_1^{N_1}(x)]) - E[\hat{J}_0^{N_0}(x)] \leq O\left(\frac{\ln N_0}{N_0}\right), x \in X. \tag{3.13}$$

and

$$E[\hat{J}_1^{N_1}(x)] \geq \mathcal{T}_1(E[\hat{J}_2^{N_2}(x)]) - O\left(\frac{\ln N_1}{N_1}\right), x \in X. \tag{3.14}$$

Applying the $\mathcal{T}_0$-operator to both sides of (3.14) and using the monotonicity property of $\mathcal{T}_t$ (see, e.g., [13]), we have

$$\mathcal{T}_0(E[\hat{J}_1^{N_1}(x)]) \geq \mathcal{T}_0(\mathcal{T}_1(E[\hat{J}_2^{N_2}(x)])) - O\left(\frac{\ln N_1}{N_1}\right), x \in X. \tag{3.15}$$

Therefore, combining (3.13) and (3.15) yields

$$\mathcal{T}_0(\mathcal{T}_1(E[\hat{J}_2^{N_2}(x)])) - E[\hat{J}_0^{N_0}(x)] \leq O\left(\frac{\ln N_0}{N_0} + \frac{\ln N_1}{N_1}\right), x \in X.$$

Repeating this argument yields

$$\mathcal{T}^0 \cdots \left(\mathcal{T}^T(E[\hat{J}_T^{N_T}(x)])\right) - E[\hat{J}_0^{N_0}(x)] \leq O\left(\sum_{t=0}^{T-1} \frac{\ln N_t}{N_t}\right), x \in X. \tag{3.16}$$

Observe that $\mathcal{T}^0 \cdots \left(\mathcal{T}^T(E[\hat{J}_T^{N_T}(x)])\right) = J_0^*(x), x \in X$. Rewriting Equation (3.16), we finally have

$$0 \leq J_0^*(x) - E[\hat{J}_0^{N_0}(x)] \leq O\left(\sum_{t=0}^{T-1} \frac{\ln N_t}{N_t}\right), x \in X,$$

where the first inequality above follows because $J^*(x)$ is the true optimal reward-to-go. ∎

Remark 3.3.1 *Note that the assumption $R_{\max} \leq \frac{1}{T}$ can be relaxed by adding a scaling factor to the upper confidence bound in (3.5), i.e.,*

$$\max_{a \in A}\left(\hat{Q}_t(x,a) + R_{\max} T \sqrt{\frac{2 \ln \bar{n}}{N_{a,t}^x}}\right).$$

It can be shown that the result in Theorem 3.3.1 (cf. [8]) now becomes

$$\rho(n) \leq \sum_{a: Q(x,a) < J(x)} \left[\frac{8 R_{\max}^2 T^2 \ln n}{J(x) - Q(x,a)} + (1 + \frac{\pi^2}{3})(J(x) - Q(x,a))\right].$$

It is also easy to verify that all convergence results (including convergence rate) in this Chapter still hold for this modification.

3.4 A Numerical Example

We now apply the AMS algorithm to a classical finite horizon inventory control problem with lost sales. In these problems, the inventory level is periodically reviewed, orders are placed and received, demand is realized, and the new inventory level for the period is calculated, on which costs are charged. The objective is to find the (in general non-stationary) policy to minimize expected costs, which comprise holding, order, and penalty costs. In here, demand is assumed to be a discrete random variable.

Let D_t denote the demand in period t, x_t the inventory level at the end of period t (which is the inventory at the beginning of period $t+1$), a_t the order amount in period t, p the per period per unit demand lost penalty cost, h the per period per unit inventory holding cost, K the fixed (set-up) cost per order, and L the maximum inventory level (storage capacity), i.e., $x_t \in \{0, 1, \ldots, L\}$. Then the dynamics of the inventory level evolves as follows:

$$x_{t+1} = (x_t + a_t - D_t)^+ .$$

The objective function is the expectation of the total cost given by

$$\sum_{t=1}^{T} \left[h(x_t + a_t - D_t)^+ + p(D_t - x_t - a_t)^+ + K \cdot I\{a_t > 0\} \right],$$

where x_0 is the initial inventory level, T is the number of periods (time horizon), and $I\{\cdot\}$ is the indicator function. Note that we are ignoring per-unit order costs for simplicity.

We consider two versions: (i) fixed order amount q; (ii) any (integral) order amount (up to capacity). In both cases, if the order amount would bring the inventory level above the inventory capacity M, then that order cannot be placed, i.e., that order amount action is not feasible in that state. In case (i), there are just two actions (order or no order), whereas in case (ii), the number of actions depends on the capacity limit.

Central to the context of the algorithm is that the underlying distribution is unknown, and that only samples are available. Furthermore, there is no structural knowledge on the form of the optimal policy. However, the example selected here was chosen to be simple in order to allow for the optimal solution to be solved easily by standard techniques once the distribution is given, so that the performance of the algorithm could be evaluated.

In actual implementation, a slight modification is required for this example, because it is a minimization problem, whereas AMS was written for a maximization problem. Conceptually, the most straightforward way is just to take the reward as the negative of the cost function. Equivalently, we change (3.5) in AMS by replacing the "max" operator with the "min" operator and the addition with subtraction, i.e.,

$$\min_{a \in A} \left(\hat{Q}_t(x,a) - \sqrt{\frac{2 \ln \bar{n}}{N^x_{a,t}}} \right).$$

With $K = 0$ (no fixed order cost), the optimal order policy is easily solvable without dynamic programming, because the periods are decoupled, and the problem reduces to solving a single-period inventory optimization problem. In case (i), the optimal policy follows a threshold rule, in which an order is placed if the inventory is below a certain level; otherwise, no order is placed. The threshold (order point) is given by

$$s = \min_{x \geq 0} \{x : hE[(x+q-D)^+] + pE[(D-q-x)^+] \geq hE[(x-D)^+] + pE[(D-x)^+]\},$$

i.e., one orders in period t if $x_t < s$ (assuming that $x_t + q \leq L$; also, if the set is empty, then take $s = \infty$, i.e., an order will always be placed). In case (ii), the problem becomes a newsboy problem, with a base-stock (order up to) solution given by

$$S = F^{-1}(p/(p+h)),$$

i.e., one orders $(S - x_t)^+$ for in period t (with the implicit assumption that $S \leq L$).

For the $K > 0$ case (i), the optimal policy is again a threshold (order point) policy, but the order point is nonstationary, whereas in case (ii), the optimal policy is of the (s, S) type, again non stationary. To obtain the true solutions, standard backwards induction was employed, using knowledge of the underlying demand distribution.

For the numerical experiments, we used the following parameter settings: horizon $T = 3$; capacity $L = 20$; initial inventory $x_1 = 5$; demand $D_t \sim DU(0, 9)$ (discrete uniform); holding cost $h = 1$; penalty cost $p = 1$ and $p = 10$; fixed order cost $K = 0$ and $K = 5$; fixed order amount for case (i): $q = 10$. Note that since the order quantity is greater than the maximum demand for our values of the parameters, i.e., $q > D_t$ always, placing an order guarantees no lost sales.

3.4.1 Two Alternative Estimators

Preliminary experiments with the algorithm indicated relatively slow convergence, so we decided to consider alternative estimators to improve the empirical performance. But first, we present a theorem, which will be useful in studying these estimators.

Theorem 3.4.1 *Let $\{X_i,\ i = 1, 2, \ldots\}$ be a sequence of i.i.d. random variables with $0 \le X_i \le D$ and $E[X_i] = \mu\ \forall\ i$, and let M be a bounded integer-valued random variable, with $0 \le M \le K$ for some positive integer K. If the event $\{M = n\}$ is independent of $\{X_{n+1}, X_{n+2}, \ldots\}$, then for any given $\varepsilon > 0$ and $n \in \mathcal{Z}^+$, we have*

$$P\big(\big|\frac{1}{M}\sum_{i=1}^{M} X_i - \mu\big| \ge \varepsilon, M \ge n\big) \le 2e^{-n(\tau\varepsilon - \Lambda_D(\tau)D^2)} \quad \forall\ \tau \in (0, \tau_{max}), \tag{3.17}$$

where τ_{max} satisfies $\tau_{max} \ne 0$ and $1 + (D + \varepsilon)\tau_{max} - e^{D\lambda_{max}} = 0$ (see Figure 3.4), and $\Lambda_D(\tau) := \frac{e^{D\lambda} - 1 - \tau D}{D^2}$.

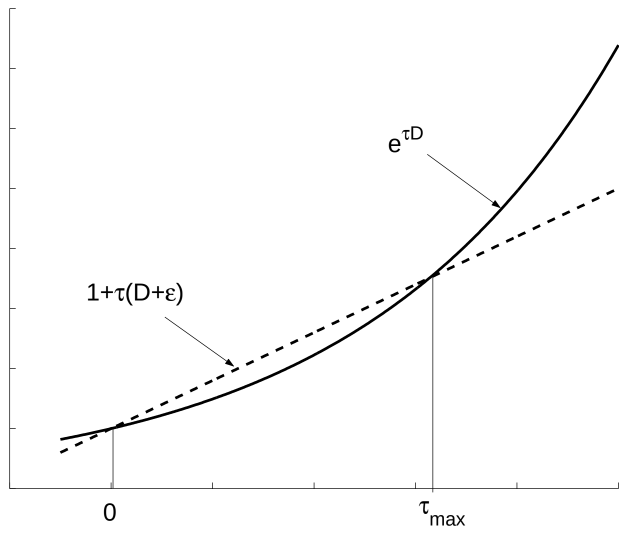


Figure 3.4: A sketch of the function $f_1(\tau) = e^{\tau D}$ and the function $f_2(\tau) = 1 + \tau(D + \varepsilon)$.

Proof: Let $Y_k = \sum_{i=1}^{k}(X_i - \mu)$. It is easy to see that the sequence $\{Y_k\}$ forms a martingale. Therefore, for any $\tau > 0$,

$$\begin{aligned} P\Big(\frac{1}{M}\sum_{i=1}^{M} X_i - \mu \geq \varepsilon, M \geq n\Big) &= P(Y_M \geq M\varepsilon, M \geq n), \\ &= P(\tau Y_M - \Lambda_D(\tau)\langle Y\rangle_M \geq \tau M\varepsilon - \Lambda_D(\tau)\langle Y\rangle_M, M \geq n), \end{aligned}$$

where

$$\langle Y\rangle_n = \sum_{j=1}^{n} E[(\Delta Y_j)^2|\mathcal{F}_{j-1}],\ \Delta Y_j = Y_j - Y_{j-1},$$

and $\mathcal{F}_j$ is the σ-field generated by $\{Y_1, \ldots, Y_j\}$.

Now for any $\tau \in (0, \tau_{max})$, and for any $n_1 \geq n_0$, where $n_0,\ n_1 \in \mathcal{Z}^+$, and $\mathcal{Z}^+$ is the set of all positive integers, we have

$$\begin{aligned} \tau(n_1 - n_0)\varepsilon &\geq \frac{e^{D\lambda} - 1 - \tau D}{D^2}(n_1 - n_0)D^2 \\ &\geq \Lambda_D(\tau)\Big[\sum_{j=1}^{n_1} E[(\Delta Y_j)^2|\mathcal{F}_{j-1}] - \sum_{j=1}^{n_0} E[(\Delta Y_j)^2|\mathcal{F}_{j-1}]\Big], \end{aligned}$$

which implies that

$$\tau n_1\varepsilon - \Lambda_D(\tau)\langle Y\rangle_{n_1} \geq \tau n_0\varepsilon - \Lambda_D(\tau)\langle Y\rangle_{n_0},\ \ \forall\ \tau \in (0, \tau_{max}).$$

Thus for all $\tau \in (0, \tau_{max})$,

$$\begin{aligned} P\Big(\frac{1}{M}\sum_{i=1}^{M} X_i - \mu \geq \varepsilon, M \geq n\Big) &\leq P(\tau Y_M - \Lambda_D(\tau)\langle Y\rangle_M \geq \tau n\varepsilon - \Lambda_D(\tau)\langle Y\rangle_n, M \geq n), \\ &\leq P(\tau Y_M - \Lambda_D(\tau)\langle Y\rangle_M \geq \tau n\varepsilon - \Lambda_D(\tau) n D^2, M \geq n), \\ &= P(e^{\tau Y_M - \Lambda_D(\tau)\langle Y\rangle_M} \geq e^{\tau n\varepsilon - n\Lambda_D(\tau)D^2}, M \geq n). \quad (3.18) \end{aligned}$$

It can be shown that (cf. e.g., Lemma 1 in [77], pp. 505) the sequence $\{Z_t(\tau) = e^{\tau Y_t - \Lambda_D(\tau)\langle Y\rangle_t},\ t \geq 1\}$ with $Z_0(\tau) = 1$ forms a non-negative supermartingale. It follows that

$$\begin{aligned} (3.18) &\leq P(e^{\tau Y_M - \Lambda_D(\tau)\langle Y\rangle_M} \geq e^{\tau n\varepsilon - n\Lambda_D(\tau)D^2}), \\ &\leq P(\sup_{0\leq t\leq K} Z_t(\tau) \geq e^{\tau n\varepsilon - n\Lambda_D(\tau)D^2}), \\ &\leq \frac{E[Z_0(\tau)]}{e^{\tau n\varepsilon - n\Lambda_D(\tau)D^2}} \text{ by maximal inequality for supermartingales (cf. [77])}, \\ &= e^{-n(\tau\varepsilon - \Lambda_D(\tau)D^2)}. \end{aligned}$$

By using a similar arguement, we can also show that

$$P\Big(\frac{1}{M}\sum_{i=1}^{M} X_i - \mu \leq -\varepsilon, M \geq n\Big) \leq e^{-n(\tau\varepsilon - \Lambda_D(\tau)D^2)}.$$

∎

Now we optimize the right-hand-side of (3.17) over τ. It is easy to verify that the optimal τ^* is given by $\tau^* = \frac{1}{D}\ln\frac{D+\varepsilon}{D} \in (0, \tau_{max})$. It follows that

$$\tau^*\varepsilon - \Lambda_D(\tau^*)D^2 = \frac{D+\varepsilon}{D}\ln\frac{D+\varepsilon}{D} - \frac{\varepsilon}{D} \approx \frac{\varepsilon^2}{D^2}, \text{ if } \varepsilon << D.$$

Therefore, we have

$$P\Big(\big|\frac{1}{M}\sum_{i=1}^{M} X_i - \mu\big| \geq \varepsilon, M \geq n\Big) \leq 2e^{-n\frac{\varepsilon^2}{D^2}}, \quad \text{if } \varepsilon << D.$$

When M is deterministic, this result is very similar to the well-known Hoeffding's inequality [40]. Note that by (3.17), we also have

$$\sum_{n=0}^{\infty} P\big(\big|\frac{1}{M}\sum_{i=1}^{M} X_i - \mu\big| \geq \varepsilon, M \geq n\big) \leq \frac{2e^{-(\tau\varepsilon - \Lambda_D(\tau))D^2}}{1 - e^{-(\tau\varepsilon - \Lambda_D(\tau))D^2}} < \infty \;\; \forall\, \tau \in (0, \tau_{max}),$$

which in turn, by the Borel-Cantelli lemma, implies that the event $\big\{\big|\frac{1}{M}\sum_{i=1}^{M} X_i - \mu\big| \geq \varepsilon, M \geq n\big\}$ will only happen finitely often w.p.1 as $n \to \infty$. And since ε is arbitrary, we have

$$\frac{1}{M}\sum_{i=1}^{M} X_i \to \mu \;\text{ w.p.1 as } M \to \infty. \tag{3.19}$$

Now consider an estimator that chooses the action that is sampled the most in order to estimate the value function, i.e., for $t < T$,

$$\begin{aligned} \tilde{J}_t^{N_t}(x) &= \tilde{Q}_t(x, a_t^*), \text{ where } a_t^* = \arg\max_a \{N_{a,t}^x\}, \\ \tilde{Q}_t(x,a) &= R_t(x,a) + \alpha \frac{1}{N_{a,t}^x} \sum_{y \in \Lambda_a^x} \tilde{J}_{t+1}^{N_{t+1}}(y). \end{aligned} \tag{3.20}$$

Now we show that if the one-stage-cost function $R(x,a)$ is *deterministic*, this estimator underestimates the optimal value function w.p.1. in an asymptotic sense. At the final stage T, we clearly have $\tilde{J}_T^{N_T}(x) = J_T^*(x) = 0, \;\forall\, x$. Thus, at stage $t = T-1$, we have

$$\tilde{J}_{T-1}^{N_{T-1}}(x) = R_{T-1}(x, a_{T-1}^*) \leq \max_a R_{T-1}(x,a) = J_{T-1}^*(x), \;\forall\, x.$$

It follows that

$$\begin{aligned} \tilde{J}_{T-2}^{N_{T-2}}(x) &= R_{T-2}(x, a_{T-2}^*) + \alpha \frac{1}{N_{a_{T-2}^*, T-2}^x} \sum_{y \in \Lambda_{a_{T-2}^*}^x} \tilde{J}_{T-1}^{N_{T-1}}(y) \\ &\leq R_{T-2}(x, a_{T-2}^*) + \alpha \frac{1}{N_{a_{T-2}^*, T-2}^x} \sum_{y \in \Lambda_{a_{T-2}^*}^x} J_{T-1}^*(y) \\ &\leq R_{T-2}(x, a_{T-2}^*) + \alpha E[J_{T-1}^*(Y)|x, a_{T-2}^*] + \alpha \Delta_{T-2}^* \\ &= J_{T-2}^*(x) + \alpha \Delta_{T-2}^*, \;\forall\, x, \end{aligned}$$

where Y is a random variable distributed according to $P_{x,\cdot|T-2}(a^*_{T-2})$, and $\Delta^*_{T-2} := \max_{x,a}\left\{\frac{1}{N^x_{a,T-2}}\sum_{y\in\Lambda^x_a} J^*_{T-1}(y) - E[J^*_{T-1}(Y)|x,a]\right\}$.

Thus, by an inductive argument, it is easy to see that

$$\tilde{J}^{N_0}_0(x) \le J^*_0(x) + \sum_{t=0}^{T-2}\alpha^{t+1}\Delta^*_t. \tag{3.21}$$

Thus, by taking the limit at both sides of (3.21) and using (3.19),

$$\lim_{N_t\to\infty\ \forall t}\tilde{J}^{N_0}_0(x) \le \lim_{N_t\to\infty\ \forall t}\left\{J^*_0(x) + \sum_{t=0}^{T-2}\alpha^{t+1}\Delta^*_t\right\} = J^*_0(x) \quad \text{w.p.1},$$

since $N^x_{a,t}\to\infty$ as $N_t\to\infty\ \forall\ a$.

We combine it with the original estimator to obtain the following estimator:

$$\bar{J}^{N_t}_t(x) = \max\left\{\tilde{Q}_t(x,a^*), \sum_{a\in A}\frac{N^x_{a,t}}{N_t}\tilde{Q}_t(x,a)\right\}. \tag{3.22}$$

Intuitively, the reason behind combining via the max operator is that the estimator would be choosing the best between the two possible estimators of the Q-function, so the new estimator will at least share the same convergence rate as the original estimator.

A second alternative estimator replaces the weighted sum of the Q-value estimates in (3.7) by the maximum of the estimates, i.e., for $t < T$,

$$\hat{J}^{N_t}_t(x) = \max_{a\in A}\hat{Q}_t(x,a). \tag{3.23}$$

For the *non-adaptive* case, it can be shown that this estimator is also asymptotically unbiased, has an upward finite-sample bias for maximization problems and downward finite-sample bias for minimization problems such as the inventory control problem. Actually, it turns out that by using a similar argument as above, we can in fact establish the probability one convergence of this estimator.

At the final stage $t = T$, $\hat{J}_T^{N_T}(x) = J_T^*(x) = 0$, $\forall\ x$. Thus, when $t = T-1$, $\hat{J}_{T-1}^{N_{T-1}}(x) = \max_a R_{T-1}(x,a) = J_{T-1}^*(x)$, $\forall\ x$. When $t = T-2$, we have,

$$\begin{aligned}
\hat{J}_{T-2}^{N_{T-2}}(x) &= \max_a \left\{ R_{T-2}(x,a) + \alpha \frac{1}{N_{a,T-2}^x} \sum_{y\in\Lambda_a^x} \hat{J}_{T-1}^{N_{T-1}}(y) \right\} \\
&= \max_a \left\{ R_{T-2}(x,a) + \alpha \frac{1}{N_{a,T-2}^x} \sum_{y\in\Lambda_a^x} J_{T-1}^*(y) \right\} \\
&= \max_a \Bigg\{ R_{T-2}(x,a) + \alpha E\big[J_{T-1}^*(Y)|x,a\big] \\
&\qquad + \alpha \left[\frac{1}{N_{a,T-2}^x} \sum_{y\in\Lambda_a^x} J_{T-1}^*(y) - E\big[J_{T-1}^*(Y)|x,a\big] \right] \Bigg\} \\
&\le \max_a \big\{ R_{T-2}(x,a) + \alpha E\big[J_{T-1}^*(Y)|x,a\big] \big\} + \alpha \Delta_{T-2}^+ \\
&= J_{T-2}^*(x) + \alpha \Delta_{T-2}^+ \quad \forall\ x,
\end{aligned}$$

where $\Delta_{T-2}^+ := \max_{x,a} \big\{ \frac{1}{N_{a,T-2}^x} \sum_{y\in\Lambda_a^x} J_{T-1}^*(y) - E\big[J_{T-1}^*(Y)|x,a\big] \big\}$. Thus, by the monotonicity of the dynamic programming algorithm, we have

$$\hat{J}_0^{N_0}(x) \le J_0^*(x) + \sum_{t=0}^{T-2} \alpha^{t+1} \Delta_t^+.$$

A similar argument can also be used to show that

$$J_0^*(x) - \sum_{t=0}^{T-2} \alpha^{t+1} \Delta_t^- \le \hat{J}_0^{N_0}(x) \le J_0^*(x) + \sum_{t=0}^{T-2} \alpha^{t+1} \Delta_t^+, \tag{3.24}$$

where $\Delta_t^- := \max_{x,a} \big\{ E\big[J_{t+1}^*(Y)|x,a\big] - \frac{1}{N_{a,t}^x} \sum_{y\in\Lambda_a^x} J_{t+1}^*(y) \big\}$. Hence, since both the state and action spaces are finite, the probability one convergence of the estimator follows by taking limit at both sides of (3.24) and then using (3.19).

3.4.2 Numerical Results

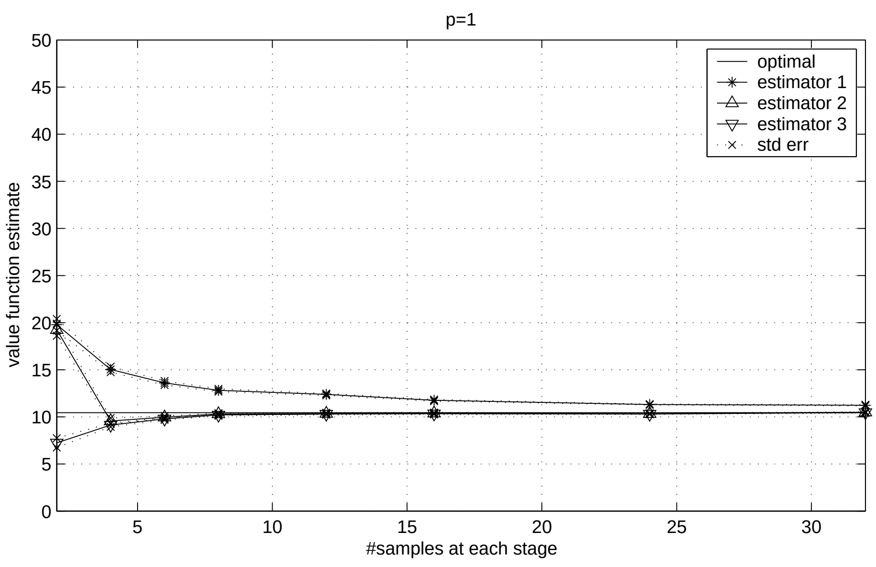
Figures 3.5, 3.6, 3.7, and 3.8 show the convergence of the estimates as a function of the number of samples at each stage for each of the respective cases (i) and (ii) considered. In each figure, estimator 1 stands for the original estimator using (3.7), and estimators 2 and 3 refer to the estimators using $\bar{J}(x)$ from (3.22) and $\hat{J}(x)$ from (3.23), respectively. Tables 3.1 and 3.2 give the performances of these estimators for each of the respective cases (i) and (ii), including the optimal value and policy parameters. The results indicate the convergence of all three estimators. We see that the two alternative estimators provide superior empirical performance than the original estimator, we believe that this is because the original estimator uses the weighted sum of the Q-function estimates, which could be too conservative for test cases where greedy estimators may have better performances.

3.5 Concluding Remarks

The AMS algorithm targets MDPs with relatively large state spaces; however, for problems where a relatively small set of states are likely to be revisited, it might be advantageous to store calculated values of $\hat{J}_i^{N_i}$ to avoid having to possibly recompute them, which could result in substantial savings for longer-horizon problems, since it would also avoid the costly recursive calls. The trade off in additional required storage, possibly unmanageable for very large state spaces, would have to be evaluated against the estimated resultant gains in running time.

We can extend the AMS algorithm to include the case where the reward function is random. The AMS algorithm would essentially remain identical, except that sampling would now include both the next state and the one-stage reward. However, the convergence proof is likely to require more technical manipulations. Furthermore, the assumption of bounded rewards can be relaxed by using the result in [1]. Even though the AMS algorithm will converge too in this case, unfortunately, we lose the property of the uniform logarithmic bound so that the convergence rate is expected to be very slow.

Earlier work of [18] proposed several algorithms that achieve the regret bounds of



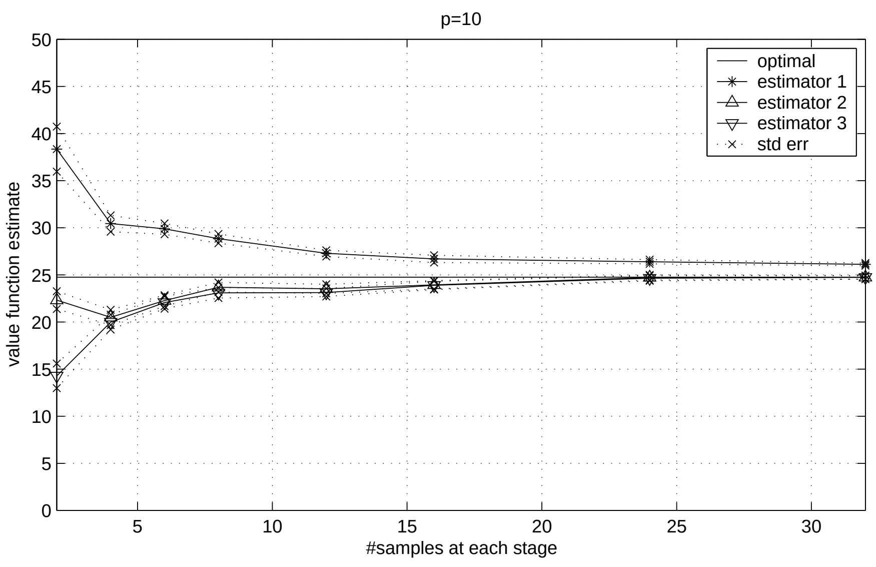
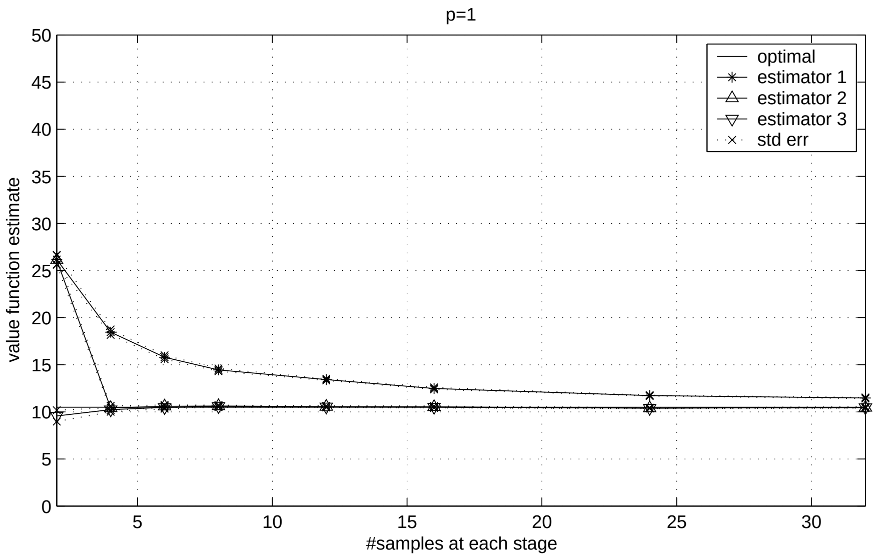


Figure 3.5: Convergence of value function estimate for the inventory control example case (i) q=10 as a function of the number of samples at each state:

$T = 3, M = 20, x_0 = 5, D_t \sim DU(0, 9), h = 1, K = 0.$

the form $c_1 + c_2 \log n + c_3 \log^2 n$, where n is the total number of plays and c_i's are positive constants not depending on n. These algorithms might also be used to create adaptive sampling algorithms for solving MDPs. However, those algorithms have the drawback



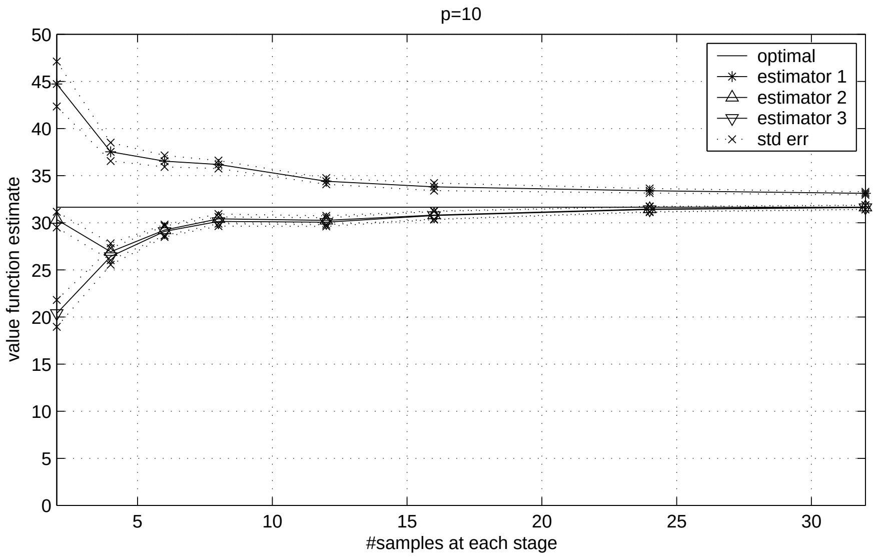
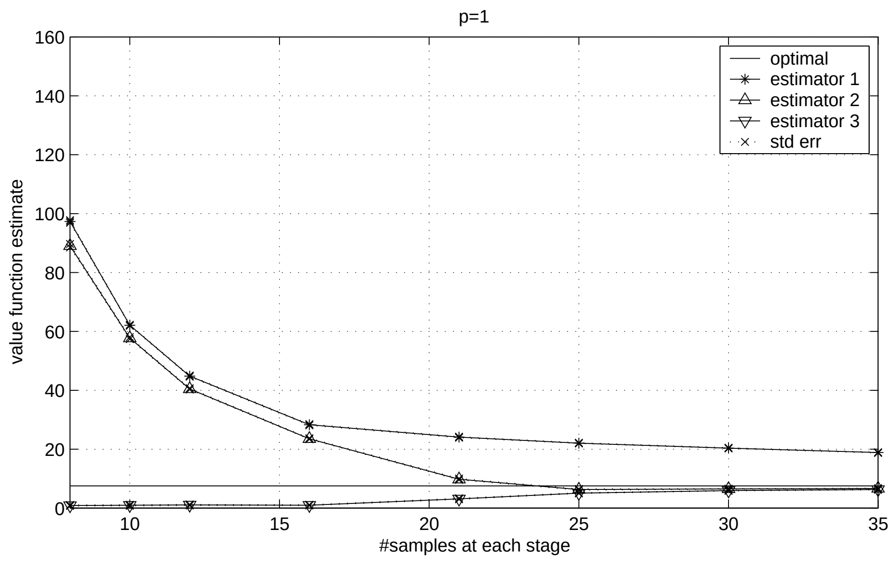


Figure 3.6: Convergence of value function estimate for the inventory control example case (i) q=10 as a function of the number of samples at each state:

$T = 3, M = 20, x_0 = 5, D_t \sim DU(0, 9), h = 1, K = 5.$

that we need to know the exact value of $\alpha(x)$ for a given state x under the assumption that not all of the actions are optimal, which is difficult to obtain in advance. This holds also for other algorithms studied in [8].



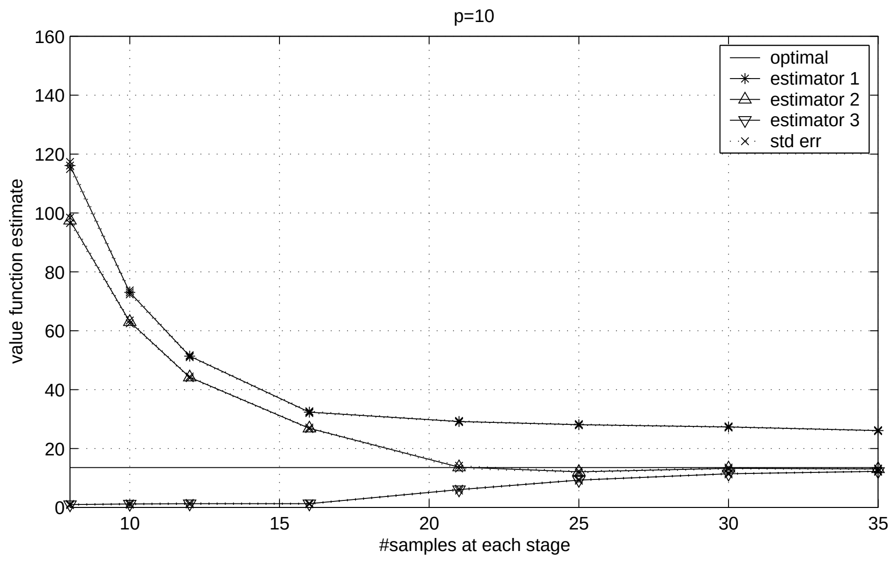
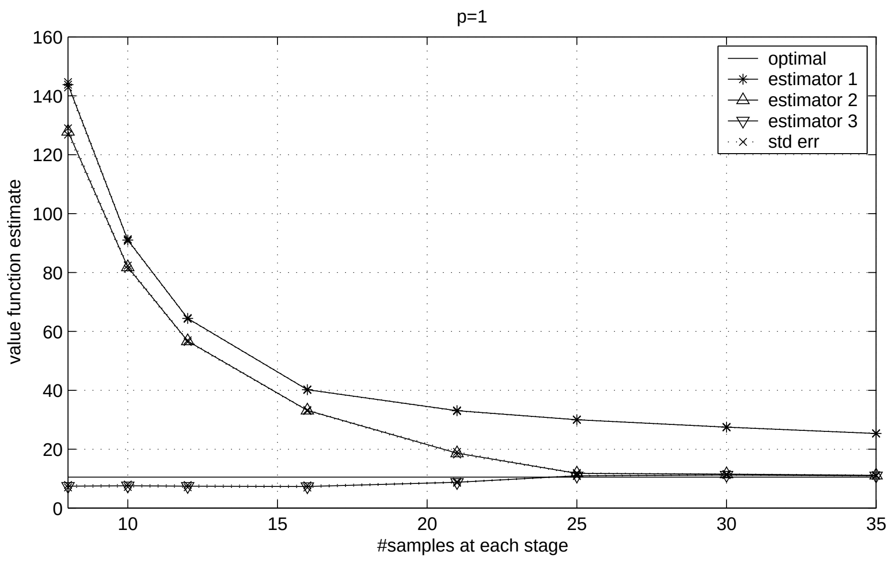


Figure 3.7: Convergence of value function estimate for the inventory control example case (ii) as a function of the number of samples at each state:

$T = 3, M = 20, x_0 = 5, D_t \sim DU(0, 9), h = 1, K = 0.$



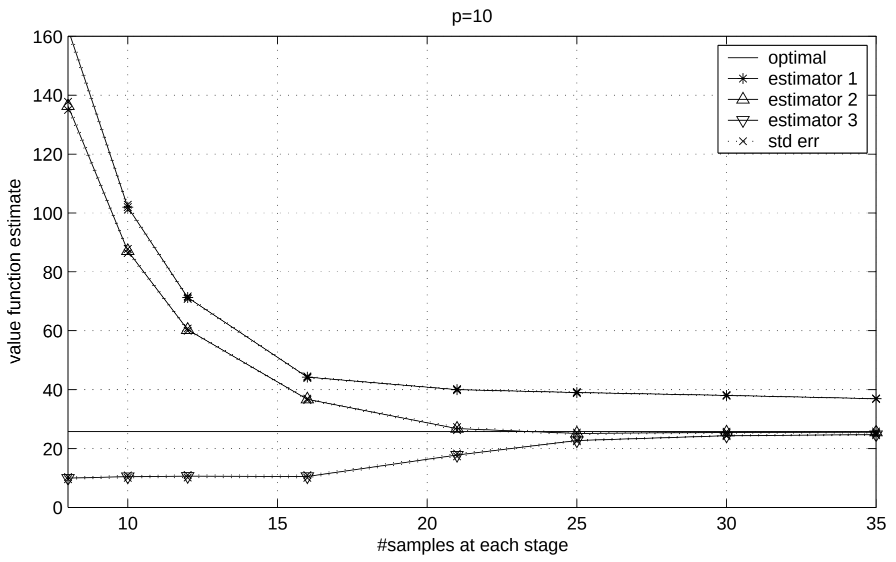


Figure 3.8: Convergence of value function estimate for the inventory control example case (ii) as a function of the number of samples at each state:

$T = 3, M = 20, x_0 = 5, D_t \sim DU(0, 9), h = 1, K = 5.$

(K,p)	optimal	N	est. 1	(std err)	est. 2	(std err)	est. 3	(std err)
	10.440	4	15.030	(0.292)	9.563	(0.322)	9.134	(0.207)
$K=0$	$s=0$	8	12.819	(0.156)	10.297	(0.096)	10.208	(0.102)
$p=1$		16	11.747	(0.093)	10.376	(0.079)	10.326	(0.081)
		32	11.227	(0.062)	10.485	(0.057)	10.450	(0.057)
	24.745	4	30.446	(0.868)	20.481	(0.817)	19.978	(0.793)
$K=0$	$s=6$	8	28.843	(0.491)	23.679	(0.515)	23.091	(0.554)
$p=10$		16	26.691	(0.382)	23.937	(0.450)	23.882	(0.437)
		32	26.118	(0.141)	24.734	(0.184)	24.728	(0.185)
	10.490	4	18.451	(0.290)	10.413	(0.223)	10.227	(0.211)
$K=5$	$s_1=0$	8	14.449	(0.154)	10.619	(0.097)	10.589	(0.095)
$p=1$	$s_2=0$	16	12.480	(0.102)	10.516	(0.096)	10.509	(0.095)
	$s_3=0$	32	11.473	(0.065)	10.458	(0.064)	10.458	(0.064)
	31.635	4	37.523	(0.980)	26.917	(0.894)	26.418	(0.883)
$K=5$	$s_1=6$	8	36.172	(0.430)	30.406	(0.508)	30.132	(0.487)
$p=10$	$s_2=6$	16	33.812	(0.399)	30.802	(0.432)	30.763	(0.431)
	$s_3=5$	32	33.113	(0.159)	31.641	(0.219)	31.617	(0.219)

Table 3.1: Value function estimate for the inventory control example case (i) as a function of the number of samples at each state: $T=3, M=20, x_0=5, D_t \sim DU(0,9), q=10, h=1$. Each entry represents the mean based on 30 independent replications (standard error in parentheses).

(K, p)	optimal	N	est. 1	(std err)	est. 2	(std err)	est. 3	(std err)
	7.500	21	24.057	(0.160)	9.793	(0.209)	3.123	(0.170)
$K = 0$	$S = 4$	25	22.050	(0.124)	6.281	(0.187)	5.063	(0.124)
$p = 1$		30	20.355	(0.114)	6.473	(0.093)	5.910	(0.089)
		35	18.823	(0.111)	6.618	(0.110)	6.263	(0.097)
	13.500	21	29.171	(0.210)	13.686	(0.463)	6.035	(0.301)
$K = 0$	$S = 9$	25	28.077	(0.208)	12.058	(0.293)	9.276	(0.230)
$p = 10$		30	27.304	(0.191)	13.277	(0.234)	11.399	(0.201)
		35	26.058	(0.164)	13.072	(0.157)	12.232	(0.176)
	10.490	21	33.047	(0.124)	18.624	(0.437)	8.727	(0.209)
$K = 5$	$s_1 = 0, S_1 = 0$	25	29.994	(0.095)	11.786	(0.158)	10.957	(0.109)
$p = 1$	$s_2 = 0, S_2 = 0$	30	27.448	(0.099)	11.516	(0.066)	11.219	(0.052)
	$s_3 = 0, S_3 = 0$	35	25.326	(0.090)	11.117	(0.068)	10.957	(0.056)
	25.785	21	39.971	(0.217)	26.760	(0.522)	17.782	(0.492)
$K = 5$	$s_1 = 6, S_1 = 9$	25	39.008	(0.191)	25.090	(0.334)	22.677	(0.263)
$p = 10$	$s_2 = 6, S_2 = 9$	30	38.029	(0.163)	25.453	(0.273)	24.345	(0.174)
	$s_3 = 6, S_3 = 9$	35	36.891	(0.116)	25.514	(0.276)	24.707	(0.230)

Table 3.2: Value function estimate for the inventory control example case (ii) as a function of the number of samples at each state: $T = 3, M = 20, x_0 = 5, D_t \sim DU(0, 9), h = 1$. Each entry represents the mean based on 30 independent replications (standard error in parentheses).

Chapter 4

An Evolutionary Random Policy Search Algorithm for Solving Infinite Horizon Markov Decision Processes with Discounted Cost

As we can see from Chapter 1.1, many current solution methods for MDP problems have concentrated on reducing the size of the state space in order to address the well-known "curse of dimensionality". However, these approaches generally require the ability to enumerate the entire action space; thus they may still be practically inefficient for problems with large action spaces. In fact, it can be seen that MDPs with large or uncountable action spaces are subject to inherent computationally intractability (cf. e.g., [71]). The reason is that the general nonlinear programming problem can be viewed as a special case of the MDP problem, thus solving general MDPs must be at least as hard as solving the general (static) multivariate nonlinear programming problems. This has motivated our research to investigate the use of different global optimization strategies to improve the performance of the current MDP solution techniques.

In this chapter, we propose an algorithm called Evolutionary Random Policy Search (ERPS) for solving infinite horizon discounted cost MDPs. The algorithm is meant to complement those highly successful state space reduction techniques introduced in Chapter 1.1. As a starting point, we will focus on MDPs where the state space is relatively small but the action space is very large, so that enumerating the entire action space becomes practically inefficient. For example, consider the problem of controlling the service rate of a single-server queue with a finite buffer size, say L, in order to minimize the average number of jobs in queue and the service cost. The state space of this problem is the possible number of jobs in the queue $\{0, 1, \ldots, L\}$, so the size of the state space is $L + 1$, whereas the possible actions might be all values on an given interval representing a service rate, in which case the action space is uncountable. From a more general point of view, if one of the aforementioned state space reduction techniques is considered (cf. Chapter 1.1), for instance, say state aggregation, then MDPs with small state spaces and large action spaces can also be regarded as the outcomes resulting from the aggregation of MDPs with large state and action spaces.

Unlike the action elimination techniques ([57], [30], cf. also Chapter 1.1), ERPS approaches the issue of large action spaces in an entirely different manner, it uses an evolutionary, population-based approach that explicitly specifies a set of good policies, and then iterates on this set to produce improving policies. The key idea is to avoid enumerating the entire action space by concentrating the search on a restricted action set at each iteration and carrying out the optimization task over the restricted set. For a given problem, ERPS proceeds iteratively by constructing and solving a sequence of sub-MDP problems, i.e., MDPs defined on smaller policy spaces. At each iteration of the algorithm, the sub-MDP constructed in the previous iteration is approximately solved by using a variant of the standard policy improvement technique, and a policy called an elite policy is generated. A group of policies is then generated based on the elite policy by using the "nearest neighbor" heuristic and random sampling of the entire action space, from which a new sub-MDP is created by restricting the original MDP problem (e.g., cost structure, transition probabilities) on the current available subsets of actions. The above steps are performed repeatedly until a specified stopping rule is satisfied. The algorithm has the property that an elite policy generated at a later generation is guaranteed to outperform (in terms of value function) the elite policy at the current generation. We show that as the number of iterations goes to infinity, the sequence of elite policies will converge with probability one to an optimal policy.

Perhaps the most straightforward and the most commonly used numerical approach in dealing with MDPs with uncountable action spaces is via the use of discretization (cf. the discussions in [72]). In practice, this could lead to computational difficulties, either resulting in an action space that is too large or in a solution that is not accurate enough. In contrast, our approach works directly on the action space, requiring no explicit discretization, and the adaptive version of the algorithm we proposed improves the efficiency of the search process and produces high quality solutions. As in standard approaches such as PI and VI, the computational complexity of each iteration of ERPS is polynomial in the size of the state space, but unlike these procedures, it is insensitive to the size of the action space, making the algorithm a promising candidate for problems with relatively small state spaces but uncountable action spaces.

4.1 Related Work

There are a few literatures applying evolutionary search methods such as genetic algorithms (GAs) for solving MDPs. Wells et al. [86] have experimented with different GA parameters (e.g., cross-over and mutation rates) for finding good limited finite memory policies for partially observable MDPs, and have discussed the effects of different GA parameters based on the empirical performance of their approach on a maze problem. Lin et al. [55] also use a GA approach to solve finite horizon partially observable MDPs, however in their approach, GA is used to construct approximations of the minimal set of affine functions that describes the value function, leading to a variant of value iteration. Barash [10] interprets the infinite horizon discounted cost MDPs as optimization problems over the policy spaces and proposes a genetic search approach that directly searches the policy space to find good stationary policies. He concludes that, by comparing with the performance of his approach with that of the standard PI, it is unlikely that policy search based on GAs can offer a competitive approach in cases where PI is implementable. More recently, Chang et al. [19] propose an algorithm called evolutionary policy iteration (EPI) to find good stationary policies for infinite horizon discounted cost MDPs with discrete state and action spaces. Their approach combines the standard procedures of GAs with the properties of infinite horizon MDPs, so that certain monotonicity property is preserved among the population of policies generated at successive iterations of the algorithm. Although their algorithm is guaranteed to converge with probability one, no performance comparisons with existing techniques are provided, and the theoretical convergence requires the action space to be finite.

ERPS shares some similarities with the EPI algorithm introduced in [19], where a sequence of "elite" policies is also produced at successive iterations of the algorithm. However, the fundamental differences are that in EPI, policies are treated as the most essential elements in optimization, and each "elite" policy is directly generated from a group of policies, whereas in our approach, policies are regarded as intermediate constructions from which sub-MDP problems are then constructed and solved; EPI follows the general framework of GAs, and thus operates only at the global level, which usually results in slow convergence. In contrast, ERPS combines global search with a local enhancement step (the "nearest neighbor" heuristic) that leads to rapid convergence once a policy is found in a small neighborhood of an optimal policy. We argue that our approach substantially improves the performance of the EPI algorithm while maintaining the computational complexity at relatively the same level.

4.2 Problem Setting

We consider the infinite horizon ($T = \infty$) MDP problem (2.1) described in Chapter 2.1 with finite state space, a general (Borel) action space, and discounted cost criterion

$$\begin{aligned} J^*(x) &= \inf_{\pi \in \Pi} J^\pi(x), \text{ and} \\ J^\pi(x) &= E\left[\sum_{t=0}^{\infty} \alpha^t R(x_t, \pi(x_t)) | x_0 = x\right], \ x \in X, \ \alpha \in (0,1), \end{aligned} \tag{4.1}$$

where throughout this chapter, unless otherwise specified, we denote the set of all stationary deterministic policies $\pi : X \to A$ by Π. Assume that there exists a stationary policy $\pi^* \in \Pi$ that achieves the optimal value $J^*(x)$ for all initial states $x \in X$, and our objective is to find such a policy. Hereafter in this chapter, we denote the size of the state space by $|X|$, and assume without lost of generality that all actions $a \in A$ are admissible for all states $x \in X$.

4.3 Algorithm Description

The basic algorithmic structure of ERPS is given in Figure 4.1, where some steps are presented only at a conceptual level. We will provide a detailed explanation of these steps and discuss their implementation details in the following subsections, where each subsection corresponds to a particular step of the algorithm.

4.3.1 Initialization

The inputs to the ERPS algorithm are an *action selection distribution* $\mathcal{P}$, an exploitation probability $q_0 \in [0,1]$, a population size $n > 1$, and a search range r_i for each state $x^i \in X$. There is a lot of flexibility in the choices of the initial population of policies, we can even take all policies in the initial population to be exactly the same. This is because of the randomized search technique employed in ERPS (cf. Chapter 4.3.3), which

Evolutionary Random Policy Search (ERPS)

- **Initialization:** Specify an *action selection distribution* $\mathcal{P}$, a population size $n > 1$, and a parameter $q_0 \in [0,1]$. For each state $x^i \in X$, $i = 1, \ldots, |X|$, specify a search range r_i. Select an initial population of policies $\Lambda_0 = \{\pi_1^0, \pi_2^0, \ldots, \pi_n^0\}$. Construct an initial sub-MDP as $\mathcal{G}_{\Lambda_0} := (X, \Gamma_0, P, R, \alpha)$, where $\Gamma_0 = \bigcup_x \Lambda_0(x)$. Set $\pi_*^{-1} := \pi_1^0$.

- **Loop until the stopping rule is satisfied:**
  - **Policy Improvement with Cost Swapping (PICS):**
    - $*$ For each $\pi_j^k \in \Lambda_k$, compute the corresponding value function $J^{\pi_j^k}$.
    - $*$ Compute the elite policy

$$\pi_*^k(x) = \arg\min_{a \in \Lambda_k(x)} \left\{ R(x,a) + \alpha \sum_y P_{x,y}(a)[\min_{\pi_j^k \in \Lambda_k} J^{\pi_j^k}(y)] \right\}, \ \forall\, x \in X.$$

  - **Construct a Sub-MDP:**

```
* for $j = 2$ to $n$
      for $i = 1$ to $|X|$
          generate a r.v. $u \sim U[0,1]$,
          if $u \le q_0$ (exploitation)
              choose an action $a$ in the neighborhood of $\pi_*^k(x^i)$ by using
              the "nearest neighbor" heuristic. Set $\pi_j^{k+1}(x^i) = a$.
          elseif $u > q_0$ (exploration)
              choose an action $a$ according to $\mathcal{P}$, set $\pi_j^{k+1}(x^i) = a$.
          end if
      end for
  end for
```

    - $*$ Set the next population of policies as $\Lambda_{k+1} = \{\pi_*^k, \pi_2^{k+1}, \ldots, \pi_n^{k+1}\}$.
    - $*$ Obtain the next sub-MDP $\mathcal{G}_{\Lambda_{k+1}} := (X, \Gamma_{k+1}, P, R, \alpha)$, where $\Gamma_{k+1} = \bigcup_x \Lambda_{k+1}(x)$.
    - $*$ Set $k \leftarrow k + 1$.

Figure 4.1: Evolutionary Random Policy Search

makes the theoretical convergence results of our approach is independent of this choice. However, to improve the performance of ERPS, we often want to maintain certain diversity among the group of policies in the initial population; one simple method to achieve such diversity is to choose each individual policy uniformly from the policy space Π (e.g.

according to a uniform distribution over the policy space).

The *action selection distribution* $\mathcal{P}$ is a prespecified probability distribution over the action space, and will be used to construct sub-MDPs (cf. Chapter 4.3.3). Note that $\mathcal{P}$ could be state dependent in general, i.e., we could prescribe for each state $x \in X$ a different action selection distribution according to some prior knowledge of the problem structure. Here, for ease of exposition, we ignore its explicit dependency on state and prescribe the same $\mathcal{P}$ for all $x \in X$. Again, one simple choice of $\mathcal{P}$ is the uniform distribution. The exploitation probability q_0 and the search range r_i will be used to construct sub-MDPs; the detailed discussion of these two parameters is deferred to Chapter 4.3.3.

4.3.2 Policy Improvement with Cost Swapping

The idea behind ERPS is to randomly split a large MDP problem into a sequence of smaller, manageable MDPs, and to extract a possibly convergent sequence of policies via solving these smaller problems. For a given population of policies $\Lambda = \{\pi_1, \pi_2, \ldots, \pi_n\}$, we consider the subsets of actions given by $\Lambda(x) := \{\pi_1(x), \pi_2(x), \ldots, \pi_n(x)\}\ \forall\, x \in X$. We can then define a sub-MDP problem $\mathcal{G}_\Lambda := (X, \Gamma, P, R, \alpha)$ by restricting the original MDP (e.g., costs, transition probabilities) on these subsets of actions, where $\Gamma := \bigcup_x \Lambda(x)$. Note that for a given state x, $\Lambda(x)$ is in general a multi-set, which may contain the same action for more than once; however, we can always discard the redundant members and view $\Lambda(x)$ as the set of admissible actions at state x. For this sub-MDP $\mathcal{G}_\Lambda$, one can of course, solve it exactly by using the PI algorithm, thus leading to a policy that improves all policies in the current population. However, it is well-known that PI is a sequential computational approach and will in general take more than one iteration to find such a policy. So instead of solving $\mathcal{G}_\Lambda$ exactly, here we propose an approach that solves it only approximately. The approach is particularly amenable to parallel computing. It manipulates the policies in a given population by combining the crossover idea in standard GAs with special MDP properties, and is able to obtain an improved policy in just one iteration. The approach consists of the following two steps and produces a policy that is superior to all of the policies in the current population we call "elite" policy.

Step 1: Obtain the value functions $J^{\pi_j},\ j = 1, \ldots, n$, by solving the equations:

$$J^{\pi_j}(x) = R(x, \pi_j(x)) + \alpha \sum_y P_{x,y}(\pi_j(x)) J^{\pi_j}(y),\ \forall\, x \in X. \tag{4.2}$$

Step 2: Compute the elite policy π_* by

$$\pi_*(x) = \arg\min_{a \in \Lambda(x)} \left\{ R(x, a) + \alpha \sum_y P_{x,y}(a) [\min_{\pi_j \in \Lambda} J^{\pi_j}(y)] \right\},\ \forall\, x \in X. \tag{4.3}$$

Since in (4.3), we are basically performing the policy improvement on the "swapped cost" $\min_{\pi_j \in \Lambda} J^{\pi_j}(x)$, we call this procedure "policy improvement with cost swapping" (PICS). PICS can be thought of as a population-based variant of the standard PI, where essentially we view each policy in a given population as a genetic material, and the way we obtain the "swapped cost" corresponds to the gene crossover in standard GAs. Note that the "swapped cost" $\min_{\pi_j \in \Lambda} J^{\pi_j}(x)$ may not be the value function corresponding to any policy; intuitively, it may prevent us from choosing a poor starting policy in the policy improvement step. We now formalize this intuition in the following theorem.

Theorem 4.3.1 *Given $\Lambda = \{\pi_1, \pi_2, \ldots, \pi_n\}$, let $\bar{J}(x) = \min_{\pi_j \in \Lambda} J^{\pi_j}(x)\ \forall\, x \in X$, and let*

$$\mu(x) = \arg\min_{a \in \Lambda(x)} \left\{ R(x, a) + \alpha \sum_y P_{x,y}(a(x)) \bar{J}(y) \right\}.$$

Then $J^\mu(x) \leq \bar{J}(x),\ \forall\, x \in X$. Furthermore, if μ is not optimal for $\mathcal{G}_\Lambda$, then $J^\mu(x) < \bar{J}(x)$ for at least one $x \in X$.

Proof: We define $J_0(x) = R(x, \mu(x)) + \alpha \sum_y P_{x,y}(\mu(x))\bar{J}(y)$, and consider the sequence $\{J_j(x),\ j = 1, 2 \ldots\}$ generated by the recursion $J_{i+1}(x) = R(x, \mu(x)) + \alpha \sum_y P_{x,y}(\mu(x))J_i(y)$, $\forall\, i = 0, 1, 2, \ldots$. At an arbitrary state x, by the definition of $\bar{J}(x)$, there exists π_j such that $\bar{J}(x) = J^{\pi_j}(x)$. It follows that

$$\begin{aligned}
J_0(x) &\leq R(x, \pi_j(x)) + \alpha \textstyle\sum_y P_{x,y}(\pi_j(x))\bar{J}(y) \\
&\leq R(x, \pi_j(x)) + \alpha \textstyle\sum_y P_{x,y}(\pi_j(x))J^{\pi_j}(y) \\
&= J^{\pi_j}(x) \\
&= \bar{J}(x)\ ,
\end{aligned}$$

and since x is arbitrary, we have

$$\begin{aligned}
J_1(x) &= R(x, \mu(x)) + \alpha \textstyle\sum_y P_{x,y}(\mu(x))J_0(y) \\
&\leq R(x, \mu(x)) + \alpha \textstyle\sum_y P_{x,y}(\mu(x))\bar{J}(y) \\
&= J_0(x)\ .
\end{aligned}$$

By induction it is easy to see that $J_{i+1}(x) \leq J_i(x)$, $\forall\, x \in X$ and $\forall\, i = 0, 1, 2, \ldots$. On the other hand, it is well known (cf. e.g., [13]) that the sequence $J_0(x), J_1(x), J_2(x), \ldots$ generated by the above recursion will converge to $J^{\mu}(x)$, $\forall\, x \in X$. Therefore we have $J^{\mu}(x) \leq \bar{J}(x)$, $\forall\, x$. Note that if $J^{\mu}(x) = \bar{J}(x)$, $\forall\, x \in X$, then PICS reduces to the standard policy improvement on policy μ, and it follows that μ satisfies the Bellman's optimality equation and is thus optimal for $\mathcal{G}_{\Lambda}$. Hence we must have $J^{\mu}(x) < \bar{J}(x)$ for some $x \in X$ whenever μ is not optimal. ∎

Now at the *kth* iteration, given the current policy population Λ_k, we compute the *kth* elite policy π_*^k via PICS. According to Theorem 4.3.1, the elite policy improves any policy in Λ_k, and since π_*^k is directly used to generate the $(k+1)th$ sub-MDP (cf. Figure 4.1 and Chapter 4.3.3), the following monotonicity property is immediately clear:

Corollary 4.3.2 *For all $k \geq 0$,*

$$J^{\pi_*^{k+1}}(x) \leq J^{\pi_*^k}(x), \quad \forall\, x \in X.$$

Proof: Follows by induction. ∎

The PICS is similar to the so-called "policy switching" proposed in [19], where an "elite" policy is also obtained at each iteration of the method. However, unlike PICS, policy switching constructs an elite policy by directly manipulating each individual policy in the population. More specifically, for the given policy population $\Lambda = \{\pi_1, \pi_2, \ldots, \pi_n\}$, the elite policy is constructed as

$$\pi_*(x) \in \left\{ \arg\min_{\pi_i \in \Lambda} (J^{\pi_i}(x))(x) \right\}, \ \forall\, x \in X, \tag{4.4}$$

where the value functions J^{π_i}, $\forall\, \pi_i \in \Lambda$ are also obtained by using the policy evaluation step, i.e., (4.2). Chang et al. [19] have shown that the elite policy π_* generated by (4.4) also improves any policy in the population Λ. Note that the computational complexity of executing (4.4) is $O(n|X|)$, which is in general much lower than the computational cost required by a direct optimization over the entire solution space.

In contrast to policy switching, PICS still retains an optimization mechanism (as in PI) over the restricted subsets of actions, which may introduce additional computational cost. However, we argue that PICS will in general substantially improve the performance of policy switching at only an extra neglectable computational expense. To illustrate this point, we now provide a intuitive comparison between these two approach; some empirical evidences can also be found later in Chapter 4.6. For a given group of policies Λ, we let Ω be the policy space induced by the sub-MDP $\mathcal{G}_\Lambda$; it is easy to see that the size of Ω is on the order of $n^{|X|}$. As we see from (4.4), policy switching only takes into account each individual policy in Λ, while PICS tends to search the entire space Ω (by carrying out an optimization over Ω), which is a much larger set than Λ. Although it is not clear in general that the elite policy generated by PICS improves the elite policy generated by policy switching, since the policy improvement step is quite fast (cf. e.g., [13]) and it focuses on the best policy updating directions, we believe this will be the case in many situations. For example, consider the case where the population Λ contains one particular policy, say $\bar{\pi}$, that dominates (in terms of value functions) all other policies in the population. It is obvious that policy switching will choose $\bar{\pi}$ as the elite policy; thus no further improvement can be achieved at the next iteration. In contrast, PICS considers the sub-MDP $\mathcal{G}_\Lambda$; as long as $\bar{\pi}$ is not optimal for $\mathcal{G}_\Lambda$ (cf. Theorem 4.3.1), a strict improving policy can always be obtained in the next iteration.

The computational complexity of each iteration of PICS is approximately the same as that of policy switching, because step 1 of PICS, i.e., (4.2), which is also used by policy switching, requires solution of n systems of linear equations, and the number of numerical operations required by using a direct method (e.g., standard Gaussian Elimination) is $O(n|X|^3)$, and this dominates the cost of step 2, which is at most $O(n|X|^2)$.

4.3.3 Sub-MDP Generation

The description of the "sub-MDP generation" step in Figure 4.1 is only at a conceptual level. To better explain this step, we now distinguish between two different settings. We start by considering the case where the action space is discrete; then we extend our discussion to the setting where the action space is continuous.

Discrete Action Spaces

By Corollary 4.3.2, the performance of the elite policy at the current iteration improves the performances of the elite policies generated at previous iterations. However, depending how new policies are generated and constructed at each iteration, strict improvement among elite policies can not always be guaranteed. Our focus now is how to achieve consistent improvements among the elite policies found at consecutive iterations. Of course, one possibility is to use unbiased random sampling and choose at each iteration a sub-MDP problem by making use of the *action selection distribution* $\mathcal{P}$. By doing so, it is obvious that we may always obtain an improved elite policy after a sufficient number of iterations. Such an unbiased sampling scheme is very effective in escaping local optima and is often useful in finding a good candidate solution. However, in practice persistent improvements will be more and more difficult to achieve as the number of iterations (sampling instances) increases, since the probability of finding better elite policies typically becomes smaller and smaller. We refer the readers to [56] for a more insightful discussion in a global optimization context. Thus, it appears that a biased sampling scheme could be more helpful.

The biased sampling scheme can be achieved in many different ways, one possibility is via the use of the "nearest neighbor" heuristic, which is the focus of our approach. To achieve a biased sampling configuration, ERPS combines exploitation ("nearest neighbor" heuristic) with exploration (unbiased sampling). The key to balance these two types of searches is the use of the exploitation probability q_0. For a given elite policy π, we construct a new policy, say $\hat{\pi}$, in the next population generation as follows: At each state $x \in X$, with probability q_0, $\hat{\pi}(x)$ is selected from a small neighborhood of $\pi(x)$; and with probability $1 - q_0$, $\hat{\pi}(x)$ is chosen according to the action selection distribution $\mathcal{P}$ (i.e., unbiased random sampling). The preceding steps are performed repeatedly until we have obtained $n - 1$ new policies, and the next population generation is simply formed by the elite policy π and the $n - 1$ newly generated policies. Intuitively, the use of exploitation will introduce more robustness into the algorithm and helps to locate the exact optimal policy, while on the other hand, the exploration step will help the algorithm to escape local optima and to find attractive policies quickly. In effect, we see that this idea is equivalent to altering the underlying *action selection distribution*, in that $\mathcal{P}$ is artificially made more peaked around the action $\pi(x)$.

To give out a detailed implementation of the "nearest neighborhood" heuristic, we should at least require that the action space A is a non-empty metric space with a defined metric on it. Once a metric $d(\cdot,\cdot)$ is given, the "nearest neighbor" heuristic in Figure 4.1 could be naturally implemented as follows:

Let r_i, a positive integer, be the search range for state x^i, $i = 1, 2, \dots, |X|$. We assume that $r_i < |A|$ for all i, where $|A|$ is the size of the action space.

- Generate a random variable l according to the discrete uniform distribution between 1 and r_i, i.e., $l \sim DU(1, r_i)$. Choose an action $\pi_j^{k+1}(x^i) = \pi(x^i) \in A$ such that under the given metric $d(\cdot,\cdot)$, $\pi(x^i)$ is the lth closest action to $\pi_*^k(x^i)$.

Remark 4.3.1 *Although the above procedure is conceptually easy, sometimes it is not easy to implement. It is often necessary to index a (possibly high-dimensional) metric space, whose complexity will depend on the dimension of the problem and the cost in evaluating the distance functions $d(\cdot,\cdot)$. However, we note that the action spaces of many MDP problems in practice are subsets of $\Re^N$, where a lot of efficient methods can be applied, such as Kd-trees ([12]) and R-trees ([38]). The most favorable situation is an action space that is "naturally ordered", e.g., in inventory control problems where actions are the number of items to be ordered $A = \{0, 1, 2, \cdots\}$, in which case the indexing and ordering becomes trivial.*

In EPI, policies in a new generation are generated by the so-called "policy mutation" procedure, which is carried out by altering a given policy in the following manner: for each state x, the currently prescribed action is replaced probabilistically. The main reason for mutating policies is to avoid being caught in a local maximum, making a probabilistic convergence guarantee possible. Two types of mutations are considered: "global mutation" and "local mutation", which are distinguished by the degree of mutation, as indicated by the number of states with changed actions in the mutated policy. The algorithm first decides whether to mutate a given policy π "globally" or "locally" according to a mutation probability P_m. Then at each state x, $\pi(x)$ is mutated with probability P_g (P_l), where P_g and P_l are the respective predefined global mutation and local mutation probabilities. It is assumed that P_g is generally close to one and P_l close to zero, thus $P_g \gg P_l$; the idea is that "global mutation" helps the algorithm to get out of local optima and "local mutation" helps the algorithm to fine-tune the solution. If a mutation is to occur, the action is changed by using the *action selection probability* $\mathcal{P}$. As a result, we see that each action in a new policy generated by "policy mutation" either remains unchanged or is altered by pure random sampling; although the so-called "local mutation" is used, no local search element is actually involved in the process. Thus, as we can see, the algorithm only operates at the global level, which is essentially equivalent to setting the exploitation probability $q_0 = 0$ in our approach.

Continuous Action Spaces

We now carry the biased sampling idea one step further by considering MDPs with continuous action spaces. We let $\mathcal{B}_A$ be the smallest σ-algebra containing all the open sets in A, and let the *action selection distribution* $\mathcal{P}$ be a probability measure defined on $(A, \mathcal{B}_A)$. Again, we assume that there is a metric $d(\cdot,\cdot)$ defined on A. Thus, a high level implementation of the exploitation step in Figure 4.1 can be described as follows:

Let $r_i > 0$ denote the search range for state x^i, $i = 1, 2, \ldots, |X|$.

- Choose an action uniformly (according to a uniform distribution) from the set of neighbors $\{a : d(a, \pi_*^k(x^i)) \leq r_i,\ a \in A\}$.

Note that in the two different action space settings we have discussed, i.e., discrete case and continuous case, the search range parameter r_i usually has different meanings. In the former case, r_i is a positive integer indicating the number of candidate actions that are the closest to the current elite action $\pi_*^k(x^i)$, whereas in the latter case, r_i is the distance from the current elite action, which may take any positive real value.

If we further impose some additional structures on A and assume that A is a non-empty open connected subset of $\Re^N$ with some metric (e.g., the infinity-norm), then a detailed implementation of the above exploitation step is as follows.

- Generate a random vector $\lambda^i = (\lambda_1^i, \ldots, \lambda_N^i)^T$ with each $\lambda_h^i \sim U[-1, 1]$ independent for all $h = 1, 2, \ldots, N$, and choose the action $\pi_j^{k+1}(x^i) = \pi_*^k(x^i) + \lambda^i r_i$.
- *If* $\pi_j^{k+1}(x^i) \notin A$, then repeat the above step.

Remark 4.3.2 *We remark that in the above implementation, the same r_i value is used along all directions of the action space. However, in practice, it is often useful to generalize r_i to a N-dimensional vector with each component controlling the search range in a particular direction of the action space.*

Remark 4.3.3 *The metric $d(\cdot, \cdot)$ used in the "nearest neighbor" heuristic implicitly imposes a structure on the action space. The efficiency of the algorithm, to a large extent, depends on how the metric is actually defined. Like most of the random search methods for global optimizations, our approach is designed to explore the structure that good* *policies tend to be clustered together. Thus, in our context, a good metric should have a good potential in representing this structure. For example, the discrete metric (i.e., $d(a,a) = 0 \ \forall\ a \in A$ and $d(a,b) = 1 \ \forall\ a,b \in A,\ a \neq b$) should never be a good choice, since it does not provide us with any useful information about the action space. For a given action space, a good metric always exists but may not be known a priori. In the special case where the action space is a subset of $\Re^N$, we take the Euclidean metric as the default metric, this is in accord with most of the optimization techniques employed in $\Re^N$.*

4.3.4 Stopping Rule

There is a lot of flexibility in the choices of stopping rules. One simple choice is to stop the algorithm when a specified maximum number of iterations is reached. We use, in the numerical experiments in Chapter 4.6, one of the most commonly used stopping rules in standard GAs (cf. e.g., [19], [79], [86]). We stop the algorithm whenever $\exists\ k > 0$, such that $\|J^{\pi_*^{k+m}} - J^{\pi_*^k}\| = 0 \ \forall\ m = 1, 2, \ldots, K$, i.e., when no further improvement in the elite policy (in terms of value function) is obtained for K consecutive iterations.

4.4 Convergence of ERPS

In this Chapter, we study the convergence properties of ERPS, in particular, we show that the sequence of elite policies generated by ERPS will converge asymptotically to an optimal policy with probability one. We start by defining some necessary notations.

For a given metric $d(\cdot,\cdot)$ on the action space A, we define the distance measure between two policies π^1 and π^2 as

$$d_\infty(\pi^1, \pi^2) := \max_{1 \le i \le |X|} d(\pi^1(x^i), \pi^2(x^i)).$$

We can now further define the σ-neighborhood ($\sigma > 0$) of a given policy $\hat{\pi} \in \Pi$ by

$$\mathcal{N}(\hat{\pi}, \sigma) := \{\pi |\ d_\infty(\hat{\pi}, \pi) \leq \sigma,\ \forall \pi \in \Pi\}.$$

For each policy $\pi \in \Pi$, we also define P_π as the transition matrix under policy π whose $(x, y)th$ entry is $P_{x,y}(\pi(x))$, and define R_π as the one-stage cost vector whose $(x)th$ entry is $R(x, \pi(x))$. Throughout the analysis, we denote by $\|\cdot\|_\infty$ the infinity-norm over $\Re^{|X|}$, given by $\|J\|_\infty := \max_{x \in X} |J(x)|$.

ERPS is randomized approach, each run of the algorithm gives a particular realization of the sequence of elite policies (i.e., a sample path); thus the algorithm induces a probability distribution over the set of all such sequences of elite policies. We denote the probability measure and expectation with respect to this distribution by $\hat{\mathcal{P}}(\cdot)$ and $\hat{E}(\cdot)$, respectively.

The convergence of ERPS is stated in the next theorem.

Theorem 4.4.1 *Let π^* be an optimal policy with corresponding value function J^{π^*}, and let the sequence of elite policies generated by ERPS together with their corresponding value functions be denoted by $\{\pi_*^k,\ k = 1, 2, \ldots\}$ and $\{J^{\pi_*^k},\ k = 1, 2, \ldots\}$, respectively. Assume that:*

1. *$q_0 < 1$.*

2. *For any given $\ell > 0$, $\mathcal{P}(\{a |\ d(a, \pi^*(x)) \leq \ell,\ a \in A\}) > 0,\ \forall x \in X$ (recall that $\mathcal{P}(\cdot)$ is a probability measure on the action space A).*

3. *There exist constants $\sigma > 0,\ \phi > 0,\ L_1 < \infty$, and $L_2 < \infty$, such that for all $\pi \in \mathcal{N}(\pi^*, \sigma)$ we have $\|P_\pi - P_{\pi^*}\|_\infty \leq \min\left\{L_1 d_\infty(\pi, \pi^*), \frac{1-\alpha}{\alpha} - \phi\right\}$ $(0 < \alpha < 1)$, and $\|R_\pi - R_{\pi^*}\|_\infty \leq L_2 d_\infty(\pi, \pi^*)$.*

Then for any given $\varepsilon > 0$, there exists a random variable $\mathcal{M}_\varepsilon > 0$ such that $\hat{\mathcal{P}}(\mathcal{M}_\varepsilon < \infty) = 1$ and $\hat{E}(\mathcal{M}_\varepsilon) < \infty$, and $\|J^{\pi_*^k} - J^{\pi^*}\|_\infty \leq \varepsilon \;\; \forall\, k \geq \mathcal{M}_\varepsilon$.

Assumption 1 restricts the exploitation probability from pure local search. Assumption 2 simply requires that any "ball" that contains the optimal policy will have a strictly positive probability measure. It is trivially satisfied if the set $\{a | d(a, \pi^*(x)) \leq \ell, \; a \in A\}$ has a positive (Borel) measure $\forall\, x \in X$ and the action selection distribution $\mathcal{P}$ has infinite tails (e.g., Gaussian, exponential). Assumption 3 imposes some Lipschitz type of conditions on P_π and R_π; it formalizes the notion that good (near-optimal) policies are clustered together, i.e., the optimal policy is not isolated (cf. Remark 4.3.3). The assumption can be straightforwardly verified if P_π and R_π are explicit functions of π, which is the case of our numerical examples in Chapter 4.6. For a given $\varepsilon > 0$, a policy π satisfying $\|J^\pi - J^{\pi^*}\|_\infty \leq \varepsilon$ is often referred to as an ε-optimal policy (cf. [13], [63]).

Remark 4.4.1 *The result in Theorem 4.4.1 implies the a.s. convergence of the sequence $\{J^{\pi_*^k}, k = 0, 1, \ldots\}$ to the optimal value function J^{π^*}. To see this, note that Theorem 4.4.1 implies that $\hat{\mathcal{P}}(\|J^{\pi_*^k} - J^{\pi^*}\|_\infty > \varepsilon) \to 0$ as $k \to \infty$ for every given ε, which means that the sequence converges in probability. Furthermore, since $\|J^{\pi_*^k} - J^{\pi^*}\|_\infty \leq \varepsilon \;\; \forall\, k \geq \mathcal{M}_\varepsilon$ is equivalent to $\sup_{\bar{k} \geq k} \|J^{\pi_*^{\bar{k}}} - J^{\pi^*}\|_\infty \leq \varepsilon \;\; \forall\, k \geq \mathcal{M}_\varepsilon$, we will also have $\hat{\mathcal{P}}(\sup_{\bar{k} \geq k} \|J^{\pi_*^{\bar{k}}} - J^{\pi^*}\|_\infty > \varepsilon) \to 0$ as $k \to \infty$, and the a.s. convergence thus follows.*

Proof: We first try to derive an upperbound for $\|J^\pi - J^{\pi^*}\|_\infty$ in terms of the distance $d_\infty(\pi, \pi^*)$. For policy π^* and policy π we have:

$$J^{\pi^*} = R_{\pi^*} + \alpha P_{\pi^*} J^{\pi^*}, \tag{4.5}$$

$$J^\pi = R_\pi + \alpha P_\pi J^\pi. \tag{4.6}$$

Now define $\Delta J^{\pi^*} = J^\pi - J^{\pi^*}$, $\Delta P_{\pi^*} = P_\pi - P_{\pi^*}$ and $\Delta R_{\pi^*} = R_\pi - R_{\pi^*}$ and subtract the above two equations. We have

$$\Delta J^{\pi^*} = [I - (I - \alpha P_{\pi^*})^{-1} \alpha \Delta P_{\pi^*}]^{-1} (I - \alpha P_{\pi^*})^{-1} (\alpha \Delta P_{\pi^*} J^{\pi^*} + \Delta R_{\pi^*}). \tag{4.7}$$

Taking the infinity-norm at both sides of (4.7) and using the consistency property of the operator norm (i.e., $\|AB\| \leq \|A\| \cdot \|B\|$), it follows that

$$\|\Delta J^{\pi^*}\|_\infty \leq \|[I-(I-\alpha P_{\pi^*})^{-1}\alpha \Delta P_{\pi^*}]^{-1}\|_\infty \|(I-\alpha P_{\pi^*})^{-1}\|_\infty (\alpha \|\Delta P_{\pi^*}\|_\infty \|J^{\pi^*}\|_\infty + \|\Delta R_{\pi^*}\|_\infty). \tag{4.8}$$

Note that assumption 3 implies $\|\Delta P_{\pi^*}\|_\infty < \frac{1-\alpha}{\alpha}$. Thus

$$\begin{aligned} \|(I - \alpha P_{\pi^*})^{-1} \alpha \Delta P_{\pi^*}\|_\infty &\leq \|(I - \alpha P_{\pi^*})^{-1}\|_\infty \alpha \|\Delta P_{\pi^*}\|_\infty \\ &< \|(I - \alpha P_{\pi^*})^{-1}\|_\infty (1 - \alpha) \\ &< 1. \end{aligned}$$

To proceed, we now distinguish between two cases, $\|J^{\pi^*}\|_\infty = 0$ and $\|J^{\pi^*}\|_\infty \neq 0$.

Case 1. If $R_{\pi^*} = 0$ (i.e., $R(x, \pi^*(x)) = 0$ for all $x \in X$), then we have $J^{\pi^*} = 0$. Thus $\Delta J^{\pi^*} = J^\pi$ and $\Delta R_{\pi^*} = R_\pi$. By noting $\|P_\pi\|_\infty = 1$, it follows from (4.6) that

$$\|\Delta J^{\pi^*}\|_\infty = \|J^\pi\|_\infty \leq \frac{1}{1 - \alpha \|P_\pi\|_\infty} \|R_\pi\|_\infty = \frac{1}{1-\alpha} \|\Delta R_{\pi^*}\|_\infty.$$

Then by assumption 3,

$$\|\Delta J^{\pi^*}\|_\infty \leq \frac{L_2}{1-\alpha} d_\infty(\pi, \pi^*). \tag{4.9}$$

Case 2. If $R_{\pi^*} > 0$ (i.e., $R(x, \pi^*(x)) > 0$ for some $x \in X$), then from (4.5), $J^{\pi^*} > 0$. Divide both sides of (4.8) by $\|J^{\pi^*}\|_\infty$, use the relation that $\|(I - B)^{-1}\| \leq \frac{1}{1-\|B\|}$ whenever

$\|B\| < 1$ and the consistency property; it immediately follows that

$$\begin{aligned}
\frac{\|\Delta J^{\pi^*}\|_\infty}{\|J^{\pi^*}\|_\infty} &\leq \frac{\|(I-\alpha P_{\pi^*})^{-1}\|_\infty}{1-\|(I-\alpha P_{\pi^*})^{-1}\|_\infty \alpha\|\Delta P_{\pi^*}\|_\infty}\left\{\alpha\|\Delta P_{\pi^*}\|_\infty + \frac{\|\Delta R_{\pi^*}\|_\infty}{\|J^{\pi^*}\|_\infty}\right\} \\
&= \frac{\|(I-\alpha P_{\pi^*})^{-1}\|_\infty\|I-\alpha P_{\pi^*}\|_\infty}{1-\|(I-\alpha P_{\pi^*})^{-1}\|_\infty \alpha\|\Delta P_{\pi^*}\|_\infty}\left\{\frac{\alpha\|\Delta P_{\pi^*}\|_\infty}{\|I-\alpha P_{\pi^*}\|_\infty} + \frac{\|\Delta R_{\pi^*}\|_\infty}{\|I-\alpha P_{\pi^*}\|_\infty\|J^{\pi^*}\|_\infty}\right\} \\
&\leq \frac{\mathcal{K}}{1-\mathcal{K}\frac{\alpha\|\Delta P_{\pi^*}\|_\infty}{\|I-\alpha P_{\pi^*}\|_\infty}}\left\{\frac{\alpha\|\Delta P_{\pi^*}\|_\infty}{\|I-\alpha P_{\pi^*}\|_\infty} + \frac{\|\Delta R_{\pi^*}\|_\infty}{\|R_{\pi^*}\|_\infty}\right\} \\
&\leq \frac{\mathcal{K}}{1-\mathcal{K}\frac{\alpha\|\Delta P_{\pi^*}\|_\infty}{\|I-\alpha P_{\pi^*}\|_\infty}}\left\{\frac{\alpha L_1}{\|I-\alpha P_{\pi^*}\|_\infty} + \frac{L_2}{\|R_{\pi^*}\|_\infty}\right\} d_\infty(\pi,\pi^*), \qquad (4.10)
\end{aligned}$$

where $\mathcal{K} = \|(I-\alpha P_{\pi^*})^{-1}\|_\infty\|I-\alpha P_{\pi^*}\|_\infty$.

In either case (see (4.9), (4.10)), we conclude that for any given $\varepsilon > 0$, there exists a $\theta > 0$ such that for any $\pi \in \mathcal{N}(\pi^*, \sigma)$ where

$$d_\infty(\pi,\pi^*) := \max_{1\leq i\leq |X|} d(\pi(x^i), \pi^*(x^i)) \leq \theta,$$

we have $\|J^\pi - J^{\pi^*}\|_\infty = \|\Delta J^{\pi^*}\|_\infty \leq \varepsilon$. Note that $\max_{1\leq i\leq |X|} d(\pi(x^i), \pi^*(x^i)) \leq \theta$ is equivalent to

$$d(\pi(x^i), \pi^*(x^i)) \leq \theta, \ \forall\ i = 1, 2, \ldots, |X|. \qquad (4.11)$$

By assumption 2, the set of actions that satisfies (4.11) will have a strictly positive probability measure, and since $q_0 < 1$, it follows that the probability a population generation does not contain a policy in the neighborhood $\mathcal{N}(\pi^*, \min\{\theta, \sigma\})$ of the optimal policy is strictly less than 1. Let ψ be the probability that a randomly constructed policy is in $\mathcal{N}(\pi^*, \min\{\theta, \sigma\})$. Then by Theorem 4.3.1, at each iteration the probability that an elite policy is obtained in $\mathcal{N}(\pi^*, \min\{\theta, \sigma\})$ is at least $1-(1-\psi)^{n-1}$, where n is the population size. Let $\mathcal{M}_\varepsilon$ denote the number of iterations required to generate such an elite policy for the first time. By the monotonicity of the sequence $\{J^{\pi_*^k},\ k = 0, 1, \ldots\}$ (cf. Corollary 4.3.2), it is clear that $\|J^{\pi_*^k} - J^{\pi^*}\|_\infty \leq \varepsilon\ \forall\ k \geq \mathcal{M}_\varepsilon$. Now consider a random variable $\bar{\mathcal{M}}$ that is geometrically distributed with a success probability of $1-(1-\psi)^{n-1}$. It is not difficult to see that $\bar{\mathcal{M}}$ dominates $\mathcal{M}_\varepsilon$ stochastically (i.e., $\bar{\mathcal{M}} \geq_{st} \mathcal{M}_\varepsilon$), and because $\psi > 0$, it follows that $\hat{E}(\mathcal{M}_\varepsilon) \leq \hat{E}(\bar{\mathcal{M}}) = \frac{1}{1-(1-\psi)^{n-1}} < \infty$. ∎

Remark 4.4.2 *In the above proof, we have used the infinity-norm. Since in finite dimensional spaces all norms are equivalent (cf. [28]), similar results can also be easily established by using different norms, e.g., the Euclidean-norm.*

Remark 4.4.3 *It should be noted that the result presented in Theorem 4.4.1 is rather theoretical, because nothing can be said about the convergence rate of the algorithm as well as how much improvement can be achieved at each iteration. As a consequence, the random variable $\mathcal{M}_\varepsilon$ could be extremely large in practice.*

Note that for a finite action space, assumption 3 in Theorem 4.4.1 is automatically satisfied, and assumption 2 also holds trivially if we take $\mathcal{P}(a) > 0$ for all actions $a \in A$. Furthermore, when the action space is finite, there always exists an $\varepsilon > 0$ such that the only ε-optimal policy is the optimal policy itself. We have the following stronger convergence result for ERPS when the action space is finite.

Corollary 4.4.2 *(Finite action space) If the action space is finite, $q_0 < 1$, and the action selection distribution $\mathcal{P}(a) > 0 \;\forall\, a \in A$, then there exists a random variable $\mathcal{M} > 0$ such that $\hat{\mathcal{P}}(\mathcal{M} < \infty) = 1$ and $\hat{E}(\mathcal{M}) < \infty$, and $J^{\pi_*^k} = J^{\pi^*} \;\forall\; k \geq \mathcal{M}$.*

4.5 Adaptive ERPS

The search range parameter r_i in ERPS is fixed throughout the algorithm. Intuitively, small search ranges concentrate the search in small regions around the desirable points and are helpful in refining promising solutions, but they often lead to small improvements in the cost function, thus slowing down the convergence process. On the other hand, large search ranges typically reduce the number of search steps needed to find a good or near optimal solution, but can be less effective in developing finer details around desirable points and may result in less accurate solutions. In this Chapter, we present a modification of the ERPS method in which the value of the search range parameter may change from one iteration to another. The idea is to adaptively shrink and expand the search range so that we can speed up the convergence process without sacrificing the solution quality. A detailed description of the adaptive ERPS is given in Figure 4.2, where we only consider the continuous action space case; the discrete action space version can be constructed similarly.

```
Adaptive ERPS

  • Initialization: Specify an initial search range $r$, parameters $K$, $1 < K_1 < K$, $K_2 > 1$, $K_3 > 1$,
    $\gamma > 1$ and a tolerance level $\varepsilon > 0$, where $K$ is the stopping control parameter as in ERPS. Set
    $\imath \leftarrow 0$, $\jmath \leftarrow 0$, and $h \leftarrow 0$.

  • while ($\imath \le K$ & $h \le K_3$)
      – Execute ERPS with search range $r$.
      – Search range update:
        if $0 < \|J^{\pi_*^{k+1}} - J^{\pi_*^k}\| \le \varepsilon$, then set $\imath \leftarrow 0$, $\jmath \leftarrow \jmath + 1$;
        elseif $\|J^{\pi_*^{k+1}} - J^{\pi_*^k}\| = 0$, then set $\imath \leftarrow \imath + 1$, $\jmath \leftarrow 0$;
        else set $\imath \leftarrow 0$, $\jmath \leftarrow 0$.
        end if

        if $\imath \ge K_1$, then set $r_{old} \leftarrow r$, $r \leftarrow r \cdot \frac{1}{\gamma}$. end if
        if $\jmath \ge K_2$, then set $r \leftarrow r \cdot \gamma$. end if
        if $r = r_{old}$, then set $h \leftarrow h + 1$; else set $h \leftarrow 0$. end if
    end while
```

Figure 4.2: Adaptive ERPS

We start by running ERPS with an initially specified search range r (for simplicity, we assume that the same search range is prescribed for all states), and monitor the performance of the elite policy obtained at each iteration. If no improvements among the elite policies are achieved for several, say K_1, consecutive iterations, then it indicates that the current search range may be too large, and we decrease it by a factor $\gamma > 1$. On the other hand, if for some consecutive iterations, say K_2, the improvements are non-zero but smaller than some given tolerance ε, then it is likely that the current search range is too small, and we increase it by γ until the improvement is greater than the specified tolerance level. The search range is updated repeatedly until it has been alternating between two values for K_3 times. Intuitively, the adaptive ERPS ensures that each improvement in the elite policy is (approximately) at least ε; when no further improvement is available either by increasing or by decreasing the search range, the value function obtained will be within distance ε of the optimal cost, i.e., the resulting elite policy is approximately ε-optimal.

Note that the validity of the ε-optimality claim relies on the assumption that if there is an improvement of at least ε available, then the algorithm will be able to find it via adaptive adjustment of the search range. The above approach retains the theoretical convergence properties of the original ERPS method and can be applied, at least in principle, to many types of action spaces as long as a metric can be specified; however, we must again emphasis that the efficiency of the approach will depend on the structure of the problem to be solved and how the underlying metric is actually defined.

4.6 Numerical Examples

In this Chapter, we investigate the empirical performance of ERPS by applying it to two discrete-time controlled queueing examples and comparing its performance with those of EPI ([19]) and standard PI. Throughout the experiment with ERPS, we use the same search range parameter value for all states, denoted by a single variable r, and choose the uniform distribution as the action selection distribution. All computational time units are in seconds.

4.6.1 A One-Dimensional Queueing Example

The following example has previously been studied in several approximate dynamic programming literatures (cf. e.g., [13], [27]). Consider a single-server queue with finite capacity, where the server can serve only one customer in a period, and the service of a customer begins/ends only at the beginning/end of any period. Assume at any period of time, there is at most one customer arrival, and arrivals at the queue are independent with probability $p = 0.2$ (i.e., no arrival with probability 0.8). The maximum queue length is $\mathcal{L}$, and an arrival that finds $\mathcal{L}$ customers in the queue is lost. We denote by x_t the state variable, be the number of customers in the system at the beginning of period t. The action (control) to be chosen at each state is the service completion probability of the server, denoted by a, which takes value in a set A. In period t, if $a(x_t)$ is chosen, then a service is completed with probability $a(x_t)$, and a cost of $R(x_t, a(x_t))$ is incurred, and resulting in a transition to state x_{t+1}. The goal is to choose the optimal service completion probability for each state such that the total infinite-horizon discounted cost $E[\sum_{t=0}^{\infty} \alpha^t R(x_t, a(x_t))]$ is minimized.

For this example, we consider two different choices of one-stage cost functions: (i) a simple function that is convex in both state and action, where the one-stage cost at any period for being in state x and taking action a is given by

$$R(x, a) = x + 50a^2;$$

(ii) a complex non-convex cost function

$$R(x,a) = x + 5\left[\frac{|X|}{2}\sin(2\pi a) - x\right]^2,$$

which induces a tradeoff in choosing between large values of a to reduce the state x and appropriate values of a to make the squared term small. Intuitively, the MDP problem resulting from case (i) may have some nice properties (e.g., free of multiple local optimal solutions), so finding an optimal solution should be a relatively easy task; whereas the cost function in case (ii) introduces some further computational difficulties (e.g., multiple local minima), intended to more fully test the effectiveness of a global algorithm like ERPS.

For both cases, unless otherwise specified, the following parameter settings are used: maximum queue length $\mathcal{L} = 48$; state space $X = \{0, 1, 2, \ldots, 49\}$; discount factor $\alpha = 0.98$; and in ERPS, population size $n = 10$, search range $r = 10$, and the standard Euclidean distance is used to define the neighborhood. All computational results for ERPS are based on 30 independent replications.

Discrete Action Space

We first take the action space to be $A = \{10^{-4}k : \ k = 0, 1, \ldots, 10^4\}$, a discretized version of the continuous interval $[0, 1]$. For this setting, we test the convergence of ERPS by varying the values of the exploitation probability. Table 4.1 gives the performance of the algorithm, where we define the relative error of a value function J by

$$\text{relerr} := \frac{\|J - J^*\|_\infty}{\|J^*\|_\infty}, \tag{4.12}$$

and J^* is the optimal value function, which is obtained by using the standard PI. The computational time required for PI to find the optimal value function J^* was 15 seconds, and the value of $\|J^*\|_\infty$ is approximately 2.32e+03. Test results clearly indicate superior performances of ERPS over PI; in particular, when $q_0 = 0.25, 0.5, 0.75$, ERPS attains the optimal solution in all 30 independent trials within 2 seconds.

q_0	stop rule (K)	Avg. time (std err)	mean relerr (std err)
	2	0.84 (0.03)	7.63e-06 (8.50e-08)
	4	1.41 (0.05)	2.78e-06 (3.29e-07)
0.0	8	2.67 (0.10)	7.83e-07 (1.06e-07)
	16	5.12 (0.16)	1.81e-07 (1.88e-08)
	32	8.91 (0.38)	6.19e-08 (1.07e-08)
	2	0.94 (0.02)	3.32e-09 (1.42e-09)
	4	1.08 (0.02)	9.65e-10 (2.59e-10)
0.25	8	1.24 (0.02)	3.02e-10 (9.51e-11)
	16	1.52 (0.03)	4.54e-11 (3.86e-11)
	32	1.85 (0.04)	0.00e-00 (0.00e-00)
	2	0.92 (0.02)	2.14e-09 (1.29e-09)
0.50	4	1.00 (0.02)	2.53e-10 (1.10e-10)
	8	1.11 (0.02)	7.61e-11 (5.02e-11)
	16	1.27 (0.03)	0.00e-00 (0.00e-00)
	2	1.14 (0.02)	4.14e-10 (2.84e-10)
0.75	4	1.19 (0.02)	2.40e-11 (1.67e-11)
	8	1.27 (0.02)	1.18e-11 (1.18e-11)
	16	1.44 (0.03)	0.00e-00 (0.00e-00)
	2	12.14 (0.02)	1.66e-10 (5.18e-11)
1.0	4	12.19 (0.02)	4.85e-11 (3.49e-11)
	8	12.28 (0.01)	0.00e-00 (0.00e-00)

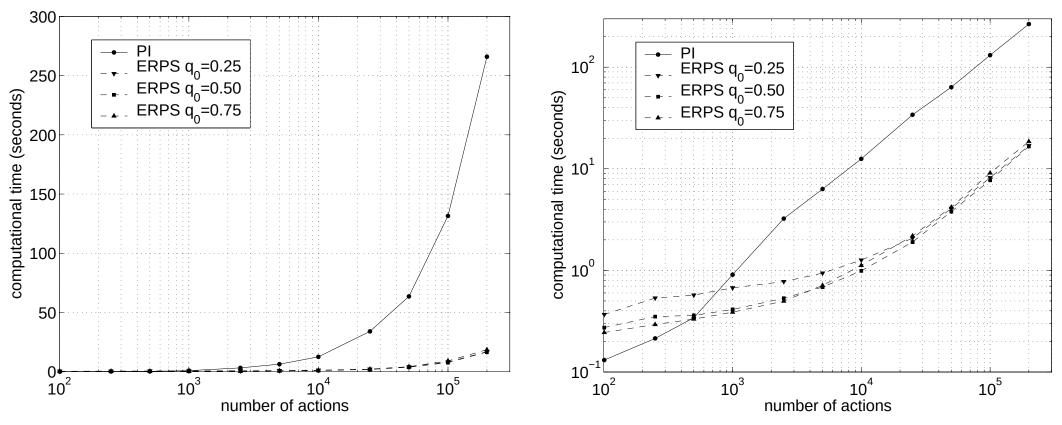
Table 4.1: Convergence results for ERPS ($n = 10$, $r = 10$) based on 30 independent replications. The standard errors are in parentheses.

To see how the computational complexity of ERPS changes with the size of the action space, we test ERPS on several MDPs with increasing numbers of actions; for each problem, the foregoing setting is used except that the action space now takes the form $A_h = \left\{hk: \ k = 0, 1, \ldots, \frac{1}{h}\right\}$, where h is the mesh size, selected sequentially (one for each problem) from the set $\left\{\frac{1}{100}, \frac{1}{250}, \frac{1}{500}, \frac{1}{1000}, \frac{1}{2500}, \frac{1}{5000}, \frac{1}{10000}, \frac{1}{25000}, \frac{1}{50000}, \frac{1}{100000}, \frac{1}{200000}\right\}$, thus the size of the action space $|A_h| = \frac{1}{h} + 1$.

We plot in Figure 4.3 the running time required for PI and ERPS to find the optimal solutions as a function of the number of actions of each MDP considered, where the results for ERPS are the averaged time over 30 independent replications. Empirical results indicate that the computational time for PI increases linearly in the number of actions (due to the requirement of enumerating the action space), while the running time required for ERPS does so in an asymptotic sense. However, ERPS significantly reduces the computational efforts of PI by roughly a factor of 14 when the size of the action space is large (number of actions greater than 10^4). We see that ERPS also delivers very competitive performances even when the action space is small. In the experiments, we used a search range $r = 10$ in ERPS, regardless of the size of the action space; we believe the performance of the algorithm could be enhanced by using a search range that is proportional to the size of the action space. Moreover, the computational effort of ERPS can be reduced considerably if we are seeking solutions within some required accuracy of the optimum rather than searching for the exact optimal solution.

For **case (ii)**, as expected, since the sine function is not monotone, the resultant MDP problem has a very high number of local minima; some typical locally optimal policies are shown in Figure 4.4.

We applied both EPI and ERPS to this case, where both algorithms start with the same initial population. The convergence of EPI and ERPS is shown in Table 4.2. The computational time required for PI to find the optimal value function J^* was 14 seconds,



(a) (b)

Figure 4.3: Running time required for PI & ERPS ($n = 10,\ r = 10$, based on 30 independent replications) to find the optimal solutions to MDPs with different numbers of actions, (a) using log-scale for horizontal axis; (b) using log-log plot.

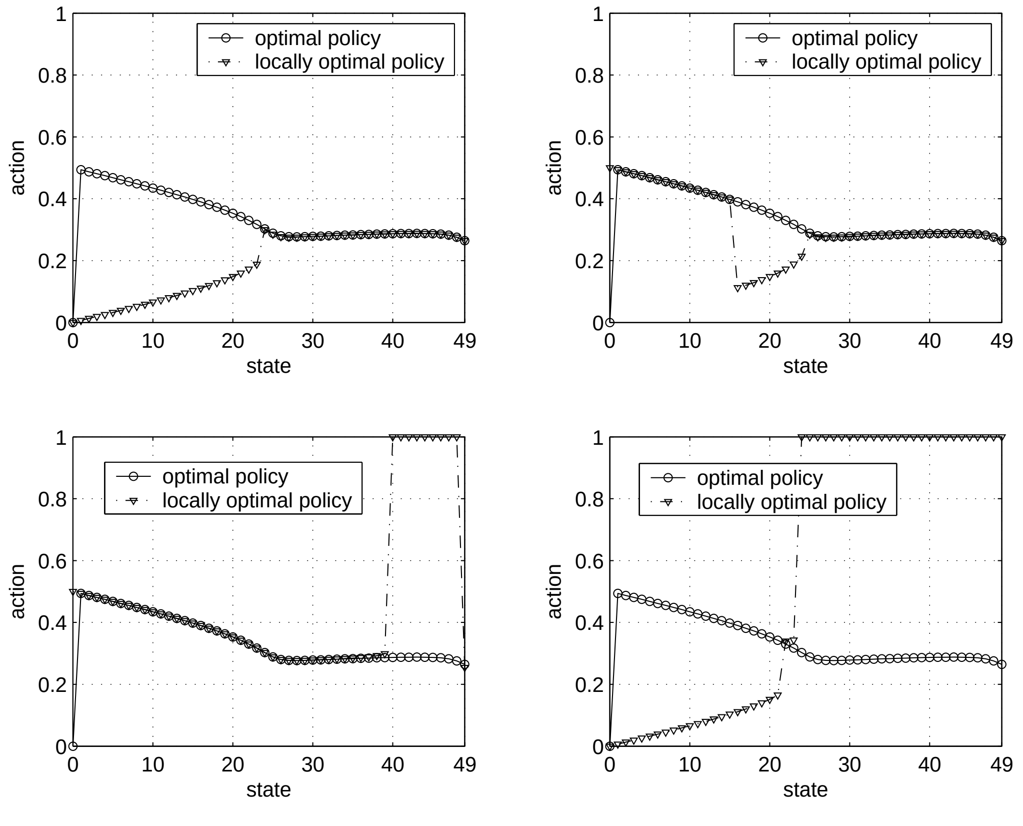


Figure 4.4: Four typical locally optimal solutions to the test problem.

and the magnitude of $\|J^*\|_\infty$ is approximately 1.03e+05. For EPI, we have tested different sets of parameters (recall from Chapter 4.3.3 that P_m is the mutation probability; and P_g (P_l) are the predefined global (local) mutation probabilities); the results reported in Table 4.2 are the best results obtained. Also note that because of the slow convergence of EPI, the values for the stopping control parameter K are chosen much larger than those for ERPS.

algorithms	stop rule (K)	Avg. time (std err)	mean relerr (std err)
EPI	20	2.13 (0.11)	1.74e-02 (1.35e-03)
$P_m = 0.1$	40	3.80 (0.16)	1.12e-02 (8.81e-04)
$P_g = 0.9$	80	6.63 (0.34)	7.13e-03 (5.37e-04)
$P_l = 0.1$	160	16.30 (0.59)	3.22e-03 (2.26e-04)
	2	1.03 (0.02)	9.81e-05 (5.17e-05)
ERPS	4	1.12 (0.03)	7.12e-05 (4.95e-05)
$q_0 = 0.5$	8	1.28 (0.03)	2.37e-05 (1.64e-05)
$r = 10$	16	1.50 (0.03)	1.06e-09 (6.59e-10)
	32	1.86 (0.04)	0.00e-00 (0.00e-00)

Table 4.2: Convergence results for EPI ($n = 10$) & ERPS ($n = 10,\ r = 10$) based on 30 independent replications. The standard errors are in parentheses.

To see how the exploitation probability q_0 affects the performance of ERPS, a set of experiments is also performed by fixing the stopping control parameter $K = 10$ and varying q_0. The numerical results are recorded in Table 4.3, where N$_{opt}$ indicates the number of times an optimal solution was found out of 30 trials. The $q_0 = 1.0$ case corresponds to pure local search. Obviously in this case, the algorithm gets trapped into a local minimum, which has a mean relative error of 5.62e-3. However, note that the standard error is zero, which means that the local minimum is estimated with very high precision. This shows that the "nearest neighbor" heuristic is indeed useful in fine-tuning the solutions. In contrast, the pure random search ($q_0 = 0$) case is helpful in escaping from the local minima, yielding a lower mean relative error of 2.59e-5, but it is not very good in locating the exact optimal solutions, as none was found out of 30 trials. Roughly, increasing q_0 between 0 and 0.5 leads to a more accurate estimation of the optimal solution; however, increasing q_0 on the range 0.6 to 1.0 decreases the quality of the solution, because the local search part begins to gradually dominate, so that the algorithm is more easily trapped in local minima. This also explains why we have larger variances when $q_0 = 0.6, 0.7, 0.8, 0.9$ in Table 4.3. Notice that the algorithm is very slow in the pure local search case; setting $q_0 < 1$ speeds up the algorithm substantially.

q_0	Avg. time (std err)	N$_{opt}$	mean relerr (std err)
0.0	3.30 (0.13)	0	2.59e-05 (6.19e-06)
0.1	1.96 (0.04)	5	4.51e-08 (8.60e-09)
0.2	1.48 (0.03)	12	1.26e-08 (3.47e-09)
0.3	1.39 (0.02)	24	2.74e-09 (2.02e-09)
0.4	1.28 (0.02)	25	2.69e-05 (1.89e-05)
0.5	1.32 (0.03)	27	8.75e-10 (6.01e-10)
0.6	1.41 (0.04)	25	6.19e-05 (3.20e-05)
0.7	1.50 (0.04)	22	1.53e-04 (6.96e-05)
0.8	1.81 (0.04)	15	3.04e-04 (7.09e-05)
0.9	2.33 (0.08)	11	7.99e-04 (1.63e-04)
1.0	7.86 (0.02)	0	5.62e-03 (0.00e-00)

Table 4.3: Performance of ERPS with different exploitation probabilities ($n = 10,\ K = 10,\ r = 10$) based on 30 independent replications. The standard errors are in parentheses.

To provide a numerical comparison between the "nearest neighbor" heuristic (biased

algorithms	parameters	Avg. time	actual relerr (std err)
ERPS $r = 10$	$q_0 = 0.0$	13.31 (0.60)	7.63e-07 (3.71e-08)
	$q_0 = 0.1$	1.20 (0.03)	4.99e-07 (5.47e-08)
	$q_0 = 0.3$	0.96 (0.04)	3.26e-07 (4.83e-08)
	$q_0 = 0.5$	0.97 (0.03)	3.84e-07 (5.08e-08)
	$q_0 = 0.7$	1.61 (0.18)	3.47e-07 (4.91e-08)
	$q_0 = 0.9$	4.03 (0.62)	2.33e-07 (4.62e-08)
ALG. 1	$P_m = 0.1,\ P_g = 0.9,\ P_l = 0.1$	62.4 (3.0)	7.61e-07 (3.67e-08)
	$P_m = 0.3,\ P_g = 0.9,\ P_l = 0.1$	33.3 (1.4)	8.42e-07 (2.76e-08)
	$P_m = 0.5,\ P_g = 0.9,\ P_l = 0.1$	26.6 (1.4)	8.35e-07 (2.93e-08)
	$P_m = 0.7,\ P_g = 0.9,\ P_l = 0.1$	22.1 (1.2)	7.88e-07 (3.34e-08)
	$P_m = 0.9,\ P_g = 0.9,\ P_l = 0.1$	20.2 (1.1)	8.44e-07 (2.55e-08)
	$P_m = 1.0,\ P_g = 1.0,\ P_l = 0.0$	17.6 (0.9)	7.67e-07 (4.08e-08)

Table 4.4: Average time required to reach a precision of at least 1.0e-6 for different algorithms. All results are based on 30 independent replications. The standard errors are in parentheses.

sampling) and the policy mutation procedure (unbiased sampling), we call the algorithm with the PICS step but policy mutation procedure as algorithm 1. In both ERPS and algorithm 1, we fix the population size $n = 10$, and stop the algorithms only when a desired accuracy is reached. In Table 4.4, we record the length of time required for different algorithms to reach a relative error of at least 1.0e-6. Indeed, we see that ERPS uses far less time to reach a required accuracy than algorithm 1 does.

Continuous Action Space

We test the algorithm when the action space A is continuous, where the service completion probability can be any value between 0 and 1. Again, two cost functions are considered, corresponding to cases (i) and (ii) in the discrete action space examples. In both cases, the maximum queue length $\mathcal{L}$, state space X, and the discount factor α are all taken to be the same as before.

In the numerical experiments, we approximated the optimal costs J_1^* and J_2^* for each of the respective cases (i) and (ii) by two value functions $\hat{J}_1^*$ and $\hat{J}_2^*$, which were computed by using the adaptive ERPS algorithm under the following parameter settings: population size $n = 10$; stopping control parameter $K = 10$; exploitation probability $q_0 = 0.5$; initial search range $r = \frac{1}{10}$; tolerance $\varepsilon = 1\text{e-}12$ for **case (i)** and $\varepsilon = 1\text{e-}10$ for **case (ii)**; $K_1 = 5$; $K_2 = 5$; $K_3 = 5$; $\gamma = 2$. We performed 200 independent runs of the adaptive ERPS algorithm for each case, and $\hat{J}_1^*$ ($\hat{J}_2^*$) was obtained as the best solution out of the 200 replications.

We set the population size $n = 10$, termination control parameter $K = 10$, and test the ERPS algorithm by using different values of the search range r. The performance of the algorithm is also compared with that of a deterministic policy iteration (PI) algorithm, where we first uniformly discretize the action space into evenly spaced points by using a mesh size h, and then apply the standard PI algorithm on the discretized problem. Tables 4.5 and 4.6 give the performances of both algorithms for cases (i) and (ii), respectively. Note that the relative errors are actually computed by replacing the optimal costs with their corresponding approximations in equation (4.12).

Test results indicate that ERPS outperforms the discretization-based PI algorithm in both cases, not only in computational time but also in solution quality. We observe that the computational time for PI increases by a factor of 2 for each halving of the mesh size, while the time for ERPS increases at a much slower rate.

algorithms	parameters	Avg. time (std err)	mean relerr (std err)
	$q_0 = 0.25$	2.54 (0.10)	1.92e-12 (3.64e-13)
ERPS	$q_0 = 0.50$	2.27 (0.09)	6.41e-13 (7.07e-14)
$(r = \frac{1}{4000})$	$q_0 = 0.75$	2.92 (0.08)	1.92e-13 (2.69e-14)
	$q_0 = 0.25$	2.61 (0.10)	4.66e-13 (6.03e-14)
ERPS	$q_0 = 0.50$	2.91 (0.10)	1.08e-13 (1.59e-14)
$(r = \frac{1}{8000})$	$q_0 = 0.75$	3.05 (0.11)	6.84e-14 (1.03e-14)
	$q_0 = 0.25$	2.84 (0.09)	1.33e-13 (2.35e-14)
ERPS	$q_0 = 0.50$	3.25 (0.10)	3.06e-14 (4.56e-15)
$(r = \frac{1}{16000})$	$q_0 = 0.75$	3.68 (0.10)	1.89e-14 (2.50e-15)
	$h = \frac{1}{4000}$	6 (N/A)	7.96e-09 (N/A)
	$h = \frac{1}{8000}$	12 (N/A)	1.72e-09 (N/A)
PI	$h = \frac{1}{16000}$	23 (N/A)	4.74e-10 (N/A)
	$h = \frac{1}{32000}$	47 (N/A)	9.52e-11 (N/A)
	$h = \frac{1}{128000}$	191 (N/A)	6.12e-12 (N/A)
	$h = \frac{1}{512000}$	781 (N/A)	3.96e-13 (N/A)

Table 4.5: Comparison of the ERPS algorithm ($n = 10,\ K = 10$) with the deterministic PI algorithm for **case (i)**. The results of ERPS are based on 30 independent replications. The standard errors are in parentheses.

4.6.2 A Two-Dimensional Queueing Example

The second example, shown in Figure 4.5, is a slight modification of the first one, with the difference being that now we have a single queue that feeds two independent servers with different service completion probabilities a_1 and a_2. We consider only the continuous action space case. The action to be chosen at each state x is $(a_1, a_2)^T$, which takes value from the set $A = [0, 1] \times [0, 1]$. We assume that an arrival that finds the system

algorithms	parameters	Avg. time (std err)	mean relerr (std err)
	$q_0 = 0.25$	2.75 (0.10)	8.49e-11 (1.50e-11)
ERPS	$q_0 = 0.50$	2.91 (0.09)	1.76e-11 (2.90e-12)
$(r = \frac{1}{4000})$	$q_0 = 0.75$	3.16 (0.09)	8.53e-12 (1.21e-12)
	$q_0 = 0.25$	3.09 (0.12)	1.70e-11 (2.57e-12)
ERPS	$q_0 = 0.50$	3.00 (0.12)	4.17e-12 (4.94e-13)
$(r = \frac{1}{8000})$	$q_0 = 0.75$	3.62 (0.08)	1.55e-12 (1.47e-13)
	$q_0 = 0.25$	3.20 (0.10)	6.08e-12 (1.17e-12)
ERPS	$q_0 = 0.50$	3.28 (0.11)	1.19e-12 (1.40e-13)
$(r = \frac{1}{16000})$	$q_0 = 0.75$	4.20 (0.12)	4.25e-13 (5.05e-14)
	$h = \frac{1}{4000}$	6 (N/A)	2.71e-07 (N/A)
	$h = \frac{1}{8000}$	11 (N/A)	5.66e-08 (N/A)
PI	$h = \frac{1}{16000}$	22 (N/A)	1.58e-08 (N/A)
	$h = \frac{1}{32000}$	43 (N/A)	5.21e-09 (N/A)
	$h = \frac{1}{128000}$	176 (N/A)	3.58e-10 (N/A)
	$h = \frac{1}{512000}$	727 (N/A)	1.71e-11 (N/A)

Table 4.6: Comparison of the ERPS algorithm ($n = 10,\ K = 10$) with the deterministic PI algorithm for **case (ii)**. The results of ERPS are based on 30 independent replications. The standard errors are in parentheses.

empty will always be served by the server with service completion probability a_1. The state space of this problem is $X = \{0, 1_{S_1}, 1_{S_2}, 2, \ldots, 48\}$, where we have assumed that the maximum queue length (no including those in service) is 46, and $1_{S_1}, 1_{S_2}$ are used to distinguish the situations whether server 1 or server 2 is busy when there is only one customer in the system. As before, the discount factor $\alpha = 0.98$.

The one-stage cost is taken to be

$$R(y, a_1, a_2) = y + \left[\frac{|X|}{2}\cos(\pi a_1) - y\right]^2 I_{\{S_1\}} + \left[\frac{|X|}{2}\sin(\pi a_2) - y\right]^2 I_{\{S_2\}},$$

where

$$I_{\{S_i\}} = \begin{cases} 1 & \text{if server } i \text{ is busy,} \\ 0 & \text{otherwise,} \end{cases} \quad (i = 1, 2), \quad \text{and} \quad y = \begin{cases} 1 & \text{if } x \in \{1_{S_1}, 1_{S_2}\}, \\ x & \text{otherwise.} \end{cases}$$

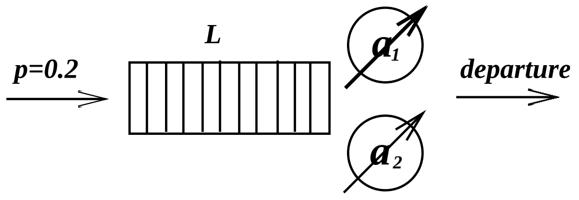


Figure 4.5: A two-dimensional queueing example.

Again, in computing the relative error, we approximated J^* by $\hat{J}^*$, which was computed by using the adaptive ERPS algorithm under the same settings (e.g., parameter settings, number of replications) as in **case (ii)** of the discrete action space examples. The value of $\|\hat{J}^*\|_\infty$ is approximately 1.72e+04.

The performances of the ERPS and the discretization-based PI are reported in Table 4.7. In ERPS, both the population size n and the stopping control parameter K are set to 10. In PI, we adopt a uniform discretization, where the same mesh size h is used in both directions of the action space. Notice that the computational time for PI increases by a factor of 4 for each halving of the mesh size, whereas the time required by ERPS increases much more slowly.

In Table 4.8, we compare the performance of the adaptive ERPS algorithm and the original ERPS algorithm in obtaining high quality solutions. In both algorithms, we choose the population size $n = 10$, the stopping control parameter $K = 10$, and the exploitation probability $q_0 = 0.5$. In adaptive ERPS, the initial search range $r = 0.1$, $\gamma = 2$, parameters

algorithms	parameters	Avg. time (std err)	mean relerr (std err)
	$q_0 = 0.25$	3.26 (0.14)	2.60e-06 (1.36e-07)
ERPS	$q_0 = 0.50$	3.20 (0.15)	1.06e-05 (9.17e-06)
$(r = \frac{1}{100})$	$q_0 = 0.75$	3.64 (0.14)	8.98e-05 (2.54e-05)
	$q_0 = 0.25$	3.37 (0.12)	6.67e-07 (3.59e-08)
ERPS	$q_0 = 0.50$	3.28 (0.12)	9.58e-06 (9.20e-06)
$(r = \frac{1}{200})$	$q_0 = 0.75$	3.89 (0.17)	9.38e-05 (2.47e-05)
	$q_0 = 0.25$	3.78 (0.11)	1.50e-07 (8.30e-09)
ERPS	$q_0 = 0.50$	3.85 (0.12)	9.30e-06 (9.21e-06)
$(r = \frac{1}{400})$	$q_0 = 0.75$	4.45 (0.14)	4.59e-05 (1.90e-05)
	$h = \frac{1}{100}$	15 (N/A)	1.65e-04 (N/A)
PI	$h = \frac{1}{200}$	57 (N/A)	4.30e-05 (N/A)
	$h = \frac{1}{400}$	226 (N/A)	8.87e-06 (N/A)

Table 4.7: A two-dimensional test example. The results of ERPS are based on 30 independent replications $(n = 10,\ K = 10)$.

K_1, K_2 and K_3 are all set to 5, and the improvements in elite policies are evaluated in the infinity-norm. We see that in order to obtain more and more accurate solutions, the search range in ERPS has to be chosen excessively small, which causes significant increase in computational effort. In contrast, the adaptive ERPS achieves better solutions within less time; moreover, the algorithm provides us with a rough estimation of the solution quality: as mentioned in Chapter 4.5, the average difference between the resultant value function J and the optimal cost J^* (i.e., $\|J - J^*\|_\infty$) will be of the same order of magnitude as ε; and the relative error can also be estimated as:

$$\text{relerr} = \frac{\|J - J^*\|_\infty}{\|J^*\|_\infty} \approx \frac{\|J - \hat{J}^*\|_\infty}{\|\hat{J}^*\|_\infty} \approx \frac{\varepsilon}{\|\hat{J}^*\|_\infty}.$$

algorithms	parameters	Avg. time	mean relerr (stderr)	$\|J - \hat{J}^*\|_\infty$ (stderr)
	$r = \frac{1}{20000}$	16.4 (0.2)	2.25e-11 (8.88e-13)	N/A (N/A)
ERPS	$r = \frac{1}{40000}$	24.8 (0.3)	5.04e-12 (1.95e-13)	N/A (N/A)
	$r = \frac{1}{80000}$	39.1 (0.5)	1.02e-12 (7.18e-14)	N/A (N/A)
Adaptive	ε =1e-07	13.8 (0.7)	9.28e-12 (3.22e-12)	1.59e-07 (5.54e-08)
ERPS	ε =1e-08	15.7 (0.8)	3.95e-13 (1.67e-13)	6.80e-09 (2.87e-09)
	ε =1e-09	17.1 (0.7)	1.09e-13 (3.12e-14)	1.87e-09 (5.37e-10)

Table 4.8: Comparison of ERPS ($n = 10$, $K = 10$, $q_0 = 0.5$) with adaptive ERPS ($n = 10$, $K = 10$, $q_0 = 0.5$, $r = 0.1$, $K_1 = K_2 = K_3 = 5$, $\gamma = 2$), based on 30 independent replications.

4.7 Conclusions and Open Problems

We presented an evolutionary, population-based method called ERPS for solving infinite horizon discounted cost MDP problems. We showed that the algorithm converges to an optimal policy w.p.1. We also illustrated the algorithm by applying it to two controlled queueing examples with large or uncountable action spaces. Numerical experiments on these small examples indicate that the ERPS algorithm is a promising approach, outperforming some existing methods (including the standard policy iteration algorithm).

Many challenges remain to be addressed before the algorithm can be applied to realistic-sized problems. The motivation behind ERPS is the setting where the action space is extremely large so that enumerating the entire action space becomes computationally impractical; however, the approach still requires enumerating the entire state space. To make it applicable to large state space problems, the algorithm will probably need to be used in conjunction with some other state space reduction techniques such as state aggregation or value function approximation. This avenue of investigation clearly merits further research.

Another important issue is the dependence of ERPS on the underlying distance metric, as determining a good metric could be challenging for those problems that do not have a natural metric already available. One possible way to get around this is to adaptively updating/changing the action selection distribution $\mathcal{P}$ at each iteration of the algorithm based on the sampling information obtained during the previous iterations. This actually constitutes a learning process; the hope is that more promising actions will have larger chances of being selected so that the future search will be biased toward the region containing high quality solutions (policies).

Another practical issue is the choice of the exploitation probability q_0. As noted earlier, the parameter q_0 serves as a tradeoff between exploitation and exploration in action selections. Preliminary experimental results indicate some robustness with respect to the value of this parameter, in that values between 0.25 and 0.75 all seem to work well; however, this may not hold for larger problems or other settings, so further investigation is required. One approach is to design a similar strategy as in simulated annealing algorithms and study the behavior of the algorithm when the value of q_0 is gradually increasing from 0 to 1, which corresponds to the transitioning of the search mechanism from pure random sampling to pure local search.

Chapter 5

A Model Reference Adaptive Search Method for Global Optimization

5.1 Introduction and Motivation

The focus of this chapter is on the development of a new randomized search framework we call model reference adaptive search (MRAS) for solving both continuous and combinatorial (deterministic) global optimization problems. Similar to what has been done in the field of machine learning and the work of [91], we characterize the existing general purpose global optimization techniques as being either *instance-based* or *model-based*, please refer to Chapter 1.2 for a review. Over the past few decades, a significant amount of research effort has been centered around classical instance-based methods. Thus, the behavior of these methods is relatively well understood. However, the model-based methods is still merely a collection of independently developed heuristic methods, without concrete theoretical foundations. The main contribution of this research is to provide a new unifying framework that addresses the most common computational difficulties faced by many model-based methods and to propose a simple way of constructing a class of model-based optimization algorithms with theoretical performance guarantee.

A schematic description of the model-based search method is given in Figure 5.1. In model-based methods, there is often an intermediate probabilistic model over the solution space, and at each iteration of these approaches new solutions are sampled/generated from the current probabilistic model; the performance of these candidate solutions are then evaluated and thus used to update the current model according to some pre-specified updating mechanism.

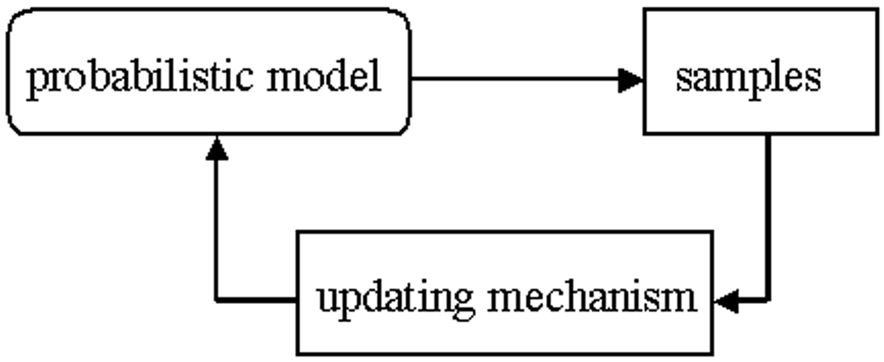


Figure 5.1: A description of the model-based methods

As we can see, there are two key questions we need to address in model-based search methods. The first question is, of course, how to update the probabilistic model. For example, traditional estimation of distribution algorithms (EDAs) (Chapter 1.2, Chapter 2.2) use an explicit construction procedure, and try to build an empirical distribution over the solution space. The updating of these empirical distributions is then usually carried out at each iteration either via measuring sample frequencies or by using the maximum likelihood estimation technique. However, the difficulty is that these empirical distributions need to be tailored to specific problems. For more complex problems, it is often tempting to use more complicated models to improve the performance of these methods, but the model construction and updating cost could be computationally expensive. Moreover, for the type of "black-box" problems, where nothing or little is known about the structure of the underlying problem, how to choose the most appropriate empirical model is a difficult issue. In contrast to the first key question, another extreme is that oftentimes one may have a nice sequence of probabilistic models, however how to sample from these distribution is a big issue. For instance, as we have discussed in Chapter 2.2, in annealing adaptive search (AAS), the majority of the computational effort is not spent in updating

Boltzmann distributions, but in how to efficiently generate samples/candidate solutions from these distributions. These fundamental issues in model-based search method are the motivation behind the MRAS method.

5.2 The Model Reference Adaptive Search Method

The Model Reference Adaptive Search method tries to address the aforementioned difficulties in the following way. A high-level description of the framework is shown in Figure 5.2, where we split the components of MRAS into two groups. The components in the red dashed box in the figure address the issue of how to sample, whereas the components in the blue box are responsible for the issue of how to update distributions. In MRAS, instead of using arbitrary (empirical) distributions (as in EDAs), we use a family of parameterized distributions as sampling distributions to generate candidate solutions. The hope is that this parameterized family is specified with some structure so that once the parameter is determined, sampling from each of these distributions should be a relatively easy task. An additional advantage by using the parameterized family is that the task of updating (empirical) sampling distributions now simplifies to the task of updating parameters associated with the distribution family. At each iteration of MRAS, the parameter is determined by minimizing certain distance between the parameterized family and an additional sequence of distributions we call *reference* distributions. These reference distributions are primarily used to guide the parameter updating process and to express the desired properties of the framework. Thus, to ensure the convergence of the framework, we often want to construct these distributions so that they will converge to a degenerated distribution concentrated only on the optimum. Intuitively, among the parameterized family, the current sampling distribution can be viewed as a compact approximation of

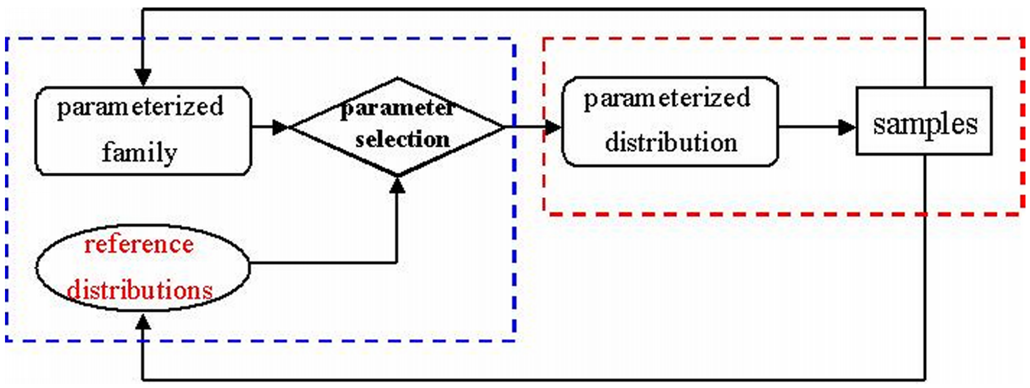


Figure 5.2: A schematic description of the MRAS framework

the reference distribution (the projection of the reference distribution on the parameterized family), and may hopefully retain some nice properties of these distributions. Thus, as the sequence of reference distributions converges, the sequence of samples generated from their compact approximations (i.e., sampling distributions) should also converge to the optimum. Since this idea is very similar to the use of reference models in adaptive control, we call this method model reference adaptive search.

5.3 The MRAS$_0$ Algorithm (Exact Version)

We consider the optimization problem introduce in Chapter 2.2:

$$x^* \in \arg\max_{x\in\mathcal{X}} H(x), \quad x \in \mathcal{X} \subseteq \Re^n, \tag{5.1}$$

where the solution space $\mathcal{X}$ is a non-empty set in $\Re^n$, and $H(\cdot) : \mathcal{X} \to \Re$ is a deterministic function that is bounded from below, i.e., $\exists \mathcal{M} > -\infty$ such that $H(x) \geq \mathcal{M}\ \forall x \in \mathcal{X}$. We will not impose any further structural (continuity, differentiability) assumptions on $H(\cdot)$. Thus, in our setting, we are interested in general optimization problems with little structure or the cases where $H(\cdot)$ does have some structures but these structures are not known as a priori. We assume that the global optimal solution to (5.1) exists and is unique, i.e., $\exists\, x^* \in \mathcal{X}$ such that $H(x) < H(x^*)\ \forall\, x \neq x^*,\ x \in \mathcal{X}$, however we note that the problem may have many locally optimal solutions.

MRAS works with a family of parameterized distribution $\{f(\cdot,\theta),\ \theta \in \Theta\}$, where Θ is the parameter space. The parameter updating in MRAS is determined by a sequence of reference distributions $\{g_k(\cdot)\}$. In particular, at each iteration k, we look at the *projection* of $g_k(\cdot)$ on the family of distributions $\{f(\cdot,\theta), \theta \in \Theta\}$ and compute the new parameter vector θ_{k+1} that minimizes the Kullback-Leibler (KL) divergence

$$\mathcal{D}(g_k, f(\cdot,\theta)) := E_{g_k}\left[\ln \frac{g_k(X)}{f(X,\theta)}\right] = \int_{\mathcal{X}} \ln \frac{g_k(x)}{f(x,\theta)} g_k(x)\nu(dx),$$

where ν is the Lebesgue/counting measure defined on $\mathcal{X}$, $X = (X_1, \ldots, X_n)$ is a random vector taking values in $\mathcal{X}$, and $E_{g_k}[\cdot]$ denotes the expectation taken with respect to $g_k(\cdot)$. Intuitively speaking, $f(\cdot,\theta_{k+1})$ can be viewed as a compact representation of the reference distribution $g_k(\cdot)$; consequently, the feasibility and effectiveness of the algorithm will, to some large extent, depend on the choices of the *reference* distributions.

As we can see from Chapter 5.2, there is a lot of flexibilities in the choices of reference distributions. So we can construct different instantiations of the framework by selecting different sequences of reference distributions. We now analyze a particular instantiation of the framework we call MRAS$_0$ by explicitly specifying a simple iterative scheme for constructing the sequence of reference distributions.

Let $g_0(x) > 0\ \forall\, x \in \mathcal{X}$ be an initial probability density/mass function (p.d.f./p.m.f.) on the solution space $\mathcal{X}$. At each iteration $k \geq 1$, we compute a new p.d.f./p.m.f. by tilting the old p.d.f./p.m.f. $g_{k-1}(x)$ with the performance function $H(x)$ (for simplicity, here we assume $H(x) > 0 \; \forall x \in \mathcal{X}$), i.e.,

$$g_k(x) = \frac{H(x) g_{k-1}(x)}{\int_{\mathcal{X}} H(x) g_{k-1}(x) \nu(dx)}, \quad \forall x \in \mathcal{X}. \tag{5.2}$$

By doing so, we are assigning more weight to solutions that have better performance. One direct consequence of this is that each iteration of (5.2) improves the expected performance. To be precise,

$$\begin{aligned} E_{g_k}[H(X)] &= \frac{E_{g_{k-1}}[(H(X))^2]}{E_{g_{k-1}}[H(X)]} \\ &\geq E_{g_{k-1}}[H(X)]. \end{aligned}$$

Furthermore, it is possible to show that the sequence $\{g_k(\cdot),\ k = 0, 1, \ldots\}$ will converge to a distribution that concentrates only on the optimal solution for arbitrary $g_0(\cdot)$. So we will have $\lim_{k\to\infty} E_{g_k}[H(X)] = H(x^*)$. The above idea has previously been used, for example, in EDAs with proportional selection schemes (cf. e.g., [90]), and in randomized algorithms for solving Markov decision processes ([20]). However, in those approaches, the construction of $g_k(\cdot)$ in (5.2) needs to be carried out *explicitly* to generate new samples; moreover, since $g_k(\cdot)$ may not have any structure, sampling from it could be computationally expensive. In MRAS, these difficulties are circumvented by projecting $g_k(\cdot)$ on the family of parameterized distributions $\{f(\cdot, \theta)\}$. On the one hand, $f(\cdot, \theta_k)$ often has some special structure and therefore could be much easier to handle, and on the other hand, the sequence $\{f(\cdot, \theta_{k+1}),\ k = 0, 1, \ldots\}$ may retain some nice properties of $\{g_k(\cdot)\}$ and also converge to a degenerate distribution concentrated on the optimal solution.

5.3.1 Algorithm Description

Throughout the analysis, we use $P_{\theta_k}(\cdot)$ and $E_{\theta_k}[\cdot]$ to denote the probability and expectation taken with respect to the p.d.f./p.m.f. $f(\cdot, \theta_k)$, and $I_{\{\cdot\}}$ to denote the indicator function, i.e.,

$$I_{\{A\}} := \begin{cases} 1 & \text{if event } A \text{ holds,} \\ 0 & \text{otherwise.} \end{cases}$$

Thus, under our notational convention,

$$P_{\theta_k}(H(X) \geq \gamma) = \int_{\mathcal{X}} I_{\{H(x)\geq\gamma\}} f(x,\theta_k)\nu(dx) \quad \text{and} \quad E_{\theta_k}[H(X)] = \int_{\mathcal{X}} H(x) f(x,\theta_k)\nu(dx).$$

Algorithm MRAS$_0$ – exact version

```
• Initialization: Specify $\rho \in (0,1]$, a small number $\varepsilon \geq 0$, a strictly increasing function
  $S(\cdot) : \Re \to \Re^+$, and an initial p.d.f./p.m.f. $f(x,\theta_0) > 0\ \forall\, x \in \mathcal{X}$. Set the iteration counter
  $k = 0$.

• Repeat until a specified stopping rule is satisfied:

    1. Calculate the $(1-\rho)$-quantile

         $\gamma_{k+1} := \sup_l \{l : P_{\theta_k}(H(X) \geq l) \geq \rho\}.$

    2. if $k = 0$, then set $\bar{\gamma}_{k+1} = \gamma_{k+1}$.
       elseif $k \geq 1$
           if $\gamma_{k+1} \geq \bar{\gamma}_k + \varepsilon$, then set $\bar{\gamma}_{k+1} = \gamma_{k+1}$.
           else set $\bar{\gamma}_{k+1} = \bar{\gamma}_k$.
           endif
       endif

    3. Compute the parameter vector $\theta_{k+1}$ as

         $\theta_{k+1} := \arg\max_{\theta\in\Theta} E_{\theta_k}\left[\frac{[S(H(X))]^k}{f(X,\theta_k)} I_{\{H(X)\geq\bar{\gamma}_{k+1}\}} \ln f(X,\theta)\right],$     (5.3)

    4. Set $k = k + 1$.
```

The MRAS$_0$ algorithm requires specification of a parameter ρ, which determines the approximate proportion of samples that will be used to update the probabilistic model. At successive iterations of the algorithm, a sequence $\{\gamma_k, k = 1, 2, \ldots\}$, i.e., the $(1-\rho)$-quantiles with respect to the sequence of p.d.f's $\{f(\cdot, \theta_k)\}$, are calculated at step 1 of MRAS$_0$. These quantile values are then used in step 2 to construct a sequence of non-decreasing thresholds $\{\bar{\gamma}_k, k = 1, 2, \ldots\}$; and only those candidate solutions that have performances better than these thresholds will be used in parameter updating (cf. equation (5.3)). As we will see, the theoretical convergence of MRAS$_0$ is unaffected by the value of the parameter ρ. The purpose of ρ in our approach is to concentrate the computational effort on the set of elite/promising samples, which is a standard technique employed in most of the population-based approaches, like GAs and EDAs.

During the initialization step of MRAS$_0$, a small number ε and a strictly increasing function $S(\cdot) : \Re \rightarrow \Re^+$ are also specified. The function $S(\cdot)$ is used to preserve the correct performance order among candidate solutions and to account for the cases where the values of $H(x)$ are negative for some x, and the parameter ε ensures that each strict increment in the sequence $\{\bar{\gamma}_k\}$ is lower bounded, i.e.,

$$\inf_{\substack{\bar{\gamma}_{k+1} \neq \bar{\gamma}_k \\ k=1,2,\ldots}} (\bar{\gamma}_{k+1} - \bar{\gamma}_k) \geq \varepsilon.$$

We require ε to be strictly positive for continuous problems, and non-negative for discrete problems.

In continuous domains, the division by $f(x, \theta_k)$ in the performance function in step 3 is well defined if $f(x, \theta_k)$ has infinite support (e.g. normal p.d.f.), whereas in discrete/combinatorial domains, the division is still valid as long as each point x in the solution space has a positive probability of being sampled. Additional regularity conditions on $f(x, \theta_k)$ in Section 5.5 will ensure that step 3 of MRAS$_0$ can be used interchangeably with the following equation:

$$\theta_{k+1} = \arg\max_{\theta\in\Theta} \int_{x\in\mathcal{X}} [S(H(x))]^k \, I_{\{H(x)\geq\bar{\gamma}_{k+1}\}} \ln f(x,\theta)dx.$$

We now show that there is a sequence of reference models $\{g_k(\cdot),\ k=1,2,\ldots\}$ implicit in MRAS$_0$, and the parameter θ_{k+1} computed at step 3 indeed minimizes the KL-divergence $\mathcal{D}(g_{k+1}, f(\cdot,\theta))$.

Lemma 5.3.1 *The parameter θ_{k+1} computed at the kth iteration of the MRAS$_0$ algorithm minimizes the KL-divergence $\mathcal{D}\left(g_{k+1}, f(\cdot,\theta)\right)$, where*

$$g_{k+1}(x) := \frac{S(H(x))I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}g_k(x)}{E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]}\ \forall x\in\mathcal{X},\ k=1,\ldots,\ \textit{and}\ g_1(x) := \frac{I_{\{H(x)\geq\bar{\gamma}_1\}}}{E_{\theta_0}\left[\frac{I_{\{H(X)\geq\bar{\gamma}_1\}}}{f(X,\theta_0)}\right]}.$$

Proof: For brevity, define $\widehat{S}_k(H(x)) := \frac{[S(H(x))]^k}{f(x,\theta_k)}$. We have

$$g_1(x) = \frac{I_{\{H(x)\geq\bar{\gamma}_1\}}}{E_{\theta_0}\left[\frac{I_{\{H(X)\geq\bar{\gamma}_1\}}}{f(X,\theta_0)}\right]} = \frac{I_{\{H(x)\geq\bar{\gamma}_1\}}}{E_{\theta_0}\left[\widehat{S}_0(H(X))I_{\{H(X)\geq\bar{\gamma}_1\}}\right]}.$$

When $k\geq 1$, we have from the definition of $g_k(\cdot)$ above,

$$\begin{aligned} g_2(x) &= \frac{S(H(x))I_{\{H(x)\geq\bar{\gamma}_2\}}g_1(x)}{E_{g_1}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_2\}}\right]} \\ &= \frac{S(H(x))I_{\{H(x)\geq\bar{\gamma}_2\}}I_{\{H(x)\geq\bar{\gamma}_1\}}}{E_{\theta_1}\left[\widehat{S}_1(H(X))I_{\{H(X)\geq\bar{\gamma}_2\}}I_{\{H(X)\geq\bar{\gamma}_1\}}\right]} \\ &= \frac{S(H(x))I_{\{H(x)\geq\bar{\gamma}_2\}}}{E_{\theta_1}\left[\widehat{S}_1(H(X))I_{\{H(X)\geq\bar{\gamma}_2\}}\right]}, \end{aligned}$$

where the last equality follows from the fact that the sequence $\{\bar{\gamma}_k,\ k=1,2,\ldots\}$ is non-decreasing. Proceeding iteratively, it is easy to see that

$$g_{k+1}(x) = \frac{[S(H(x))]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}}{E_{\theta_k}\left[\widehat{S}_k(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]},\ \forall\ k=0,1,\ldots.$$

Thus, the KL-divergence between $g_{k+1}(\cdot)$ and $f(\cdot,\theta)$ can be written as

$$\begin{aligned} \mathcal{D}\left(g_{k+1}, f(\cdot,\theta)\right) &= E_{g_{k+1}}\left[\ln g_{k+1}(X)\right] - E_{g_{k+1}}\left[\ln f(X,\theta)\right] \\ &= E_{g_{k+1}}\left[\ln g_{k+1}(X)\right] - \frac{E_{\theta_k}\left[\widehat{S}_k(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\ln f(X,\theta)\right]}{E_{\theta_k}\left[\widehat{S}_k(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]},\ \forall\ k. \end{aligned}$$

The result follows by observing that minimizing $\mathcal{D}(g_{k+1}, f(\cdot,\theta))$ with respect to θ is equivalent to maximizing the quantity $E_{\theta_k}\left[\widehat{S}_k(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\ln f(X,\theta)\right]$. ∎

5.3.2 Global Convergence

Obviously, the convergence of the MRAS$_0$ algorithm cannot be guaranteed for an arbitrary parameterized distribution family. For example, if the parameterized family is a singleton set, (i.e., contains only one distribution), then there is in general no way to ensure the convergence of the algorithm. Another practical concern is that for an arbitrary parameterized family, the computation of the new parameter θ_{k+1} in (5.3) may not even be tractable. These suggest that we should restrict our analysis and discussions to families of distributions that exhibit some structural properties. Now we show that for a particular parameterized family called the natural exponential family (NEF), the global convergence of the algorithm can be established and the new parameter θ_{k+1} can actually be obtained analytically. We start by stating the definition of NEF and some regularity conditions.

Definition 5.3.1 *A parameterized family of p.d.f's/p.m.f's* $\{f(\cdot,\theta),\ \theta\in\Theta\subseteq\Re^m\}$ *on* $\mathcal{X}$ *is said to belong to the natural exponential family (NEF) if there exist functions* $h(\cdot):\Re^n\rightarrow\Re$, $\Gamma(\cdot):\Re^n\rightarrow\Re^m$, *and* $K(\cdot):\Re^m\rightarrow\Re$ *such that*

$$f(x,\theta)=\exp\left\{\theta^T\Gamma(x)-K(\theta)\right\}h(x),\quad \forall\,\theta\in\Theta, \tag{5.4}$$

where $K(\theta)$ *is a normalization constant, given by* $K(\theta)=\ln\int_{x\in\mathcal{X}}\exp\left\{\theta^T\Gamma(x)\right\}h(x)\nu(dx)$, *and the superscript "T" denotes the vector transposition. For the case where* $f(\cdot,\theta)$ *is a p.d.f., we assume that* $\Gamma(\cdot)$ *is a continuous mapping.*

The NEF covers a broad class of distributions like Gaussian, exponential, Poisson, binomial, geometric, and certain multivariate forms of them.

Assumptions:

A1. *For any given constant $\xi < H(x^*)$, the set $\{x : H(x) \geq \xi\} \cap \mathcal{X}$ has a strictly positive Lebesgue or discrete measure.*

A2. *For any given constant $\delta > 0$, $\sup_{x \in A_\delta} H(x) < H(x^*)$, where $A_\delta := \{x : \|x - x^*\| \geq \delta\} \cap \mathcal{X}$, and we use the convention that the supremum over the empty set to be $-\infty$.*

A3. *There exists a compact set Π such that the level set $\{x : H(x) \geq \bar{\gamma}_1\} \cap \mathcal{X} \subseteq \Pi$, where $\bar{\gamma}_1 = \sup_l \{l : P_{\theta_0}(H(X) \geq l) \geq \rho\}$ is defined as in the MRAS$_0$ algorithm.*

A4. *The maximizer of equation (5.3) is an interior point of Θ for all k.*

A5. $\sup_{\theta \in \Theta} \| \exp\{\theta^T \Gamma(x)\} \Gamma(x) h(x) \|$ *is integrable/summable with respect to x, where θ, $\Gamma(\cdot)$, and $h(\cdot)$ are defined as in Definition 5.3.1.*

Intuitively, A1 ensures that any neighborhood of the optimal solution x^* will have a positive probability of being sampled. For ease of exposition, A1 restricts the class of problems under consideration to either continuous or discrete problems; however, we remark that this work can be easily extended to problems with mixture of both continuous and discrete variables. Since $H(\cdot)$ has a unique global optimizer, A2 is satisfied by many functions encountered in practice. Note that both A1 and A2 hold trivially when $\mathcal{X}$ is (discrete) finite and the counting measure is used. Assumption A3 restricts the search of the MRAS$_0$ algorithm to some compact set; it is satisfied if the function $H(\cdot)$ has compact level sets or the solution space $\mathcal{X}$ is compact. In actual implementation of the algorithm, step 3 of MRAS$_0$ is often posed as an unconstrained optimization problem, i.e., $\Theta = \Re^m$, in which case A4 is automatically satisfied. It is also easy to verify that A5 is satisfied by most NEFs.

To show the convergence of MRAS$_0$, we will need the following key observation.

Lemma 5.3.2 *If assumptions A3−A5 hold, then we have*

$$E_{\theta_{k+1}}\left[\Gamma(X)\right] = E_{g_{k+1}}\left[\Gamma(X)\right], \quad \forall\, k = 0, 1, \ldots,$$

where $E_{\theta_{k+1}}[\cdot]$ and $E_{g_{k+1}}[\cdot]$ denote the expectations taken with respect to $f(\cdot, \theta_{k+1})$ and $g_{k+1}(\cdot)$, respectively.

Proof: Define $J_k(\theta, \bar{\gamma}_{k+1}) := \int_{\mathcal{X}} \left[S(H(x))\right]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}} \ln f(x,\theta)\nu(dx)$. Since $f(\cdot,\theta)$ belongs to the NEF, we can write

$$\begin{aligned} J_k(\theta, \bar{\gamma}_{k+1}) &= \int_{\mathcal{X}} \left[S(H(x))\right]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}} \ln h(x)\nu(dx) \\ &\quad + \int_{\mathcal{X}} \left[S(H(x))\right]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}} \theta^T\Gamma(x)\nu(dx) \\ &\quad - \int_{\mathcal{X}} \left[S(H(x))\right]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}} \ln\left[\int_{\mathcal{X}} \exp\left(\theta^T\Gamma(x)\right) h(x)\nu(dx)\right] \nu(dx). \end{aligned}$$

Thus the gradient of $J_k(\theta, \bar{\gamma}_{k+1})$ with respect to θ can be expressed as

$$\begin{aligned} \nabla_\theta J_k(\theta, \bar{\gamma}_{k+1}) &= \int_{\mathcal{X}} \left[S(H(x))\right]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}\Gamma(x)\nu(dx) \\ &\quad - \frac{\int_{\mathcal{X}} e^{\theta^T\Gamma(x)}\Gamma(x)h(x)\nu(dx)}{\int_{\mathcal{X}} e^{\theta^T\Gamma(x)}h(x)\nu(dx)} \int_{\mathcal{X}} \left[S(H(x))\right]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}\nu(dx), \end{aligned}$$

where the validity of the interchange of derivative and integral above is guaranteed by assumptions A5 and the dominated convergence theorem; see e.g., [69] for further details.

By A3 and the non-decreasing property of the sequence $\{\bar{\gamma}_k\}$, it turns out that the gradient $\nabla_\theta J_k(\theta, \bar{\gamma}_{k+1})$ is finite and thus well-defined. Moreover, since $\rho > 0$, the set $\{x : H(x) \geq \bar{\gamma}_{k+1}\} \cap \mathcal{X}$ will have a strictly positive Lebesgue/counting measure. It follows that we must have $\int_{\mathcal{X}} \left[S(H(x))\right]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}\nu(dx) > 0$.

By setting $\nabla_\theta J_k(\theta, \bar{\gamma}_{k+1}) = 0$, it immediately follows that

$$\int_{\mathcal{X}} \frac{[S(H(x))]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}\Gamma(x)}{\int_{\mathcal{X}}[S(H(x))]^k I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}\nu(dx)}\nu(dx) = \int_{\mathcal{X}} \frac{e^{\theta^T\Gamma(x)}h(x)\Gamma(x)}{\int_{\mathcal{X}} e^{\theta^T\Gamma(x)}h(x)\nu(dx)}\nu(dx),$$

and by definitions of $g_{k+1}(\cdot)$ (cf. proof of Lemma 5.3.1) and $f(\cdot,\theta)$, we have

$$E_{g_{k+1}}[\Gamma(X)] = E_{\theta}[\Gamma(X)]. \tag{5.5}$$

By assumption A4, since θ_{k+1} is the optimal solution of the problem

$$\arg\max_{\theta} J_k(\theta, \bar{\gamma}_{k+1}),$$

it must satisfy equation (5.5). Therefore we conclude that

$$E_{g_{k+1}}[\Gamma(X)] = E_{\theta_{k+1}}[\Gamma(X)], \quad \forall\, k = 0, 1, \dots.$$

∎

We have the following convergence result for the MRAS$_0$ algorithm.

Theorem 5.3.1 *Let $\{\theta_k,\ k = 1, 2, \dots\}$ be the sequence of parameters generated by MRAS$_0$. If $\varepsilon > 0$ and assumptions $A1 - A5$ are satisfied, then*

$$\lim_{k\to\infty} E_{\theta_k}\left[\Gamma(X)\right] = \Gamma(x^*), \tag{5.6}$$

where the limit is component-wise.

Remark 5.3.1 *The convergence result in Theorem 5.3.1 is much stronger than it may appear to be. For example, when $\Gamma(x)$ is a one-to-one function (which is the case for many NEFs used in practice), the convergence result (5.6) can be equivalently written as $\Gamma^{-1}\left(\lim_{k\to\infty} E_{\theta_k}\left[\Gamma(X)\right]\right) = x^*$. Also note that for some particular p.d.f.'s/p.m.f.'s, the solution vector x itself will be a component of $\Gamma(x)$ (e.g., multivariate normal distribution). Under these circumstances, we can interpret (5.6) as $\lim_{k\to\infty} E_{\theta_k}\left[X\right] = x^*$. Another special case of particular interest is when the components of the random vector $X = (X_1, \dots, X_n)$ are independent, i.e., each has a univariate p.d.f./p.m.f. of the form*

$$f(x_i, \vartheta_i) = \exp(x_i\vartheta_i - K(\vartheta_i))h(x_i), \ \vartheta_i \in \Re, \forall\, i = 1, \dots, n.$$

In this case, since the distribution of the random vector X is simply the product of the marginal distributions, we will clearly have $\Gamma(x) = x$. Thus, (5.6) is again equivalent to $\lim_{k\to\infty} E_{\theta_k}[X] = x^*$, where $\theta_k := (\vartheta_1^k, \ldots, \vartheta_n^k)$, and ϑ_i^k is the value of ϑ_i at the kth iteration.

In Lemma 5.3.2, we have already established a relationship between reference models $\{g_k(\cdot)\}$ and the sequence of sampling distributions $\{f(\cdot, \theta_k)\}$. Therefore, proving Theorem 5.3.1 amounts to showing that $\lim_{k\to\infty} E_{g_k}[\Gamma(X)] = \Gamma(x^*)$.

Proof of Theorem 5.3.1: Recall from Lemma 5.3.1 that

$$g_{k+1}(x) := \frac{S(H(x))I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}g_k(x)}{E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]} \quad \forall\, x \in \mathcal{X}, \quad k = 1, 2, \ldots.$$

Thus

$$\begin{aligned} E_{g_{k+1}}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right] &= \frac{E_{g_k}\left[[S(H(X))]^2 I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]}{E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]} \\ &\geq E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]. \end{aligned} \tag{5.7}$$

Since $\bar{\gamma}_k \leq H(x^*)\ \forall\, k$, and each strict increment in the sequence $\{\bar{\gamma}_k\}$ is lower bounded by the quantity $\varepsilon > 0$, there exists a finite $\mathcal{N}$ such that $\bar{\gamma}_{k+1} = \bar{\gamma}_k,\ \forall\, k \geq \mathcal{N}$. Before we proceed any further, we need to distinguish between two cases, $\bar{\gamma}_{\mathcal{N}} = H(x^*)$ and $\bar{\gamma}_{\mathcal{N}} < H(x^*)$.

Case 1. If $\bar{\gamma}_{\mathcal{N}} = H(x^*)$ (note that since $\rho > 0$, this could only happen when the solution space is discrete), then from the definition of $g_{k+1}(\cdot)$ (see Lemma 5.3.1), we obviously have

$$g_{k+1}(x) = 0, \quad \forall\, x \neq x^*,$$

and

$$g_{k+1}(x^*) = \frac{[S(H(x^*))]^k I_{\{H(x)=H(x^*)\}}}{\int_{\mathcal{X}}[S(H(x))]^k I_{\{H(x)=H(x^*)\}}\nu(dx)} = 1 \ \ \forall\, k \geq \mathcal{N}.$$

Hence it follows immediately that

$$E_{g_{k+1}}\left[\Gamma(X)\right] = \Gamma(x^*) \ \ \forall\, k \geq \mathcal{N}.$$

Case 2. If $\bar{\gamma}_{\mathcal{N}} < H(x^*)$, then from (5.7), we have

$$E_{g_{k+1}}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+2}\}}\right] \geq E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right], \ \ \forall\, k \geq \mathcal{N}-1, \tag{5.8}$$

i.e., the sequence $\left\{E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right], k = 1, 2, \ldots\right\}$ converges.

Now we show that the limit of the above sequence is $S(H(x^*))$. To do so, we proceed by contradiction and assume that

$$S_* := \lim_{k\to\infty} E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right] < S^* := S(H(x^*)). \tag{5.9}$$

Define the set $\mathcal{A}$ as

$$\mathcal{A} := \{x : H(x) \geq \bar{\gamma}_{\mathcal{N}}\} \cap \left\{x : S(H(x)) \geq \frac{S^* + S_*}{2}\right\} \cap \mathcal{X}.$$

Since $S(\cdot)$ is strictly increasing, its inverse $S^{-1}(\cdot)$ exists. Thus $\mathcal{A}$ can be reformulated as

$$\mathcal{A} = \left\{x : H(x) \geq \max\left\{\bar{\gamma}_{\mathcal{N}},\ S^{-1}\Big(\frac{S^* + S_*}{2}\Big)\right\}\right\} \cap \mathcal{X}.$$

And since $\bar{\gamma}_{\mathcal{N}} < H(x^*)$, $\mathcal{A}$ has a strictly positive Lebesgue/discrete measure by A1.

Notice that $g_k(\cdot)$ can be rewritten as

$$g_k(x) = \prod_{i=1}^{k-1} \frac{S(H(x))I_{\{H(x)\geq\bar{\gamma}_{i+1}\}}}{E_{g_i}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{i+1}\}}\right]} \cdot g_1(x).$$

Since $\lim_{k\to\infty} \frac{S(H(x))I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}}{E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]} = \frac{S(H(x))I_{\{H(x)\geq\bar{\gamma}_{\mathcal{N}}\}}}{S_*} > 1$, $\forall\, x \in \mathcal{A}$, we conclude that

$$\lim_{k\to\infty} g_k(x) = \infty, \quad \forall\, x \in \mathcal{A}.$$

Thus, by Fatou's lemma, we have

$$1 = \liminf_{k\to\infty} \int_{\mathcal{X}} g_k(x)\nu(dx) \geq \liminf_{k\to\infty} \int_{\mathcal{A}} g_k(x)\nu(dx) \geq \int_{\mathcal{A}} \liminf_{k\to\infty} g_k(x)\nu(dx) = \infty,$$

which is a contradiction. Hence, it follows that

$$\lim_{k\to\infty} E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right] = S^*. \tag{5.10}$$

In order to show that $\lim_{k\to\infty} E_{g_k}[\Gamma(X)] = \Gamma(x^*)$, we now bound the difference between $E_{g_k}[\Gamma(X)]$ and $\Gamma(x^*)$. Note that $\forall\, k \geq \mathcal{N}$, we have

$$\begin{aligned}\|E_{g_k}[\Gamma(X)] - \Gamma(x^*)\| &\leq \int_{\mathcal{X}} \|\Gamma(x)-\Gamma(x^*)\| g_k(x)\nu(dx) \\ &= \int_{\mathcal{C}} \|\Gamma(x)-\Gamma(x^*)\| g_k(x)\nu(dx), \end{aligned} \tag{5.11}$$

where $\mathcal{C} := \{x : H(x) \geq \bar{\gamma}_{\mathcal{N}}\} \cap \mathcal{X}$ is the support of $g_k(\cdot)$, $\forall\, k \geq \mathcal{N}$.

By the assumption on $\Gamma(\cdot)$ in Definition 5.3.1, for any given $\zeta > 0$, there exists a $\delta > 0$ such that $\|x - x^*\| < \delta$ implies $\|\Gamma(x) - \Gamma(x^*)\| < \zeta$. With A_δ defined from assumption A2, we have from (5.11),

$$\begin{aligned}\|E_{g_k}[\Gamma(X)] - \Gamma(x^*)\| &\leq \int_{A_\delta^c\cap\mathcal{C}} \|\Gamma(x)-\Gamma(x^*)\| g_k(x)\nu(dx) \\ &\quad + \int_{A_\delta\cap\mathcal{C}} \|\Gamma(x)-\Gamma(x^*)\| g_k(x)\nu(dx) \\ &\leq \zeta + \int_{A_\delta\cap\mathcal{C}} \|\Gamma(x)-\Gamma(x^*)\| g_k(x)\nu(dx), \quad \forall\, k \geq \mathcal{N}. \end{aligned} \tag{5.12}$$

The rest of the proof amounts to showing that the second term in (5.12) is also bounded. Clearly the term $\|\Gamma(x) - \Gamma(x^*)\|$ is bounded on the set $A_\delta \cap \mathcal{C}$. We only need to find a bound for $g_k(x)$.

By A2, we have

$$\sup_{x\in A_\delta\cap\mathcal{C}} H(x) \leq \sup_{x\in A_\delta} H(x) < H(x^*).$$

Define $S_\delta := S^* - S(\sup_{x\in A_\delta} H(x))$. Since $S(\cdot)$ is strictly increasing, we have $S_\delta > 0$. Thus, it follows that

$$S(H(x)) \leq S^* - S_\delta, \quad \forall\, x \in A_\delta \cap \mathcal{C}. \tag{5.13}$$

On the other hand, from (5.8) and (5.10), there exists $\bar{\mathcal{N}} \geq \mathcal{N}$ such that $\forall\, k \geq \bar{\mathcal{N}}$

$$E_{g_k}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right] \geq S^* - \frac{1}{2}S_\delta. \tag{5.14}$$

Observe that $g_k(x)$ can be alternatively expressed as

$$g_k(x) = \prod_{i=\bar{\mathcal{N}}}^{k-1} \frac{S(H(x))I_{\{H(x)\geq\bar{\gamma}_{i+1}\}}}{E_{g_i}\left[S(H(X))I_{\{H(X)\geq\bar{\gamma}_{i+1}\}}\right]} \cdot g_{\bar{\mathcal{N}}}(x), \quad \forall\, k \geq \bar{\mathcal{N}}.$$

Thus, it follows from (5.13) and (5.14) that

$$g_k(x) \leq \left(\frac{S^* - S_\delta}{S^* - S_\delta/2}\right)^{k-\bar{\mathcal{N}}} \cdot g_{\bar{\mathcal{N}}}(x), \quad \forall\, x \in A_\delta \cap \mathcal{C}, \quad \forall\, k \geq \bar{\mathcal{N}}.$$

Therefore,

$$\begin{aligned}
\|E_{g_k}[\Gamma(X)] - \Gamma(x^*)\| &\leq \zeta + \sup_{x\in A_\delta\cap\mathcal{C}} \|\Gamma(x) - \Gamma(x^*)\| \int_{A_\delta\cap\mathcal{C}} g_k(x)\nu(dx) \\
&\leq \zeta + \sup_{x\in A_\delta\cap\mathcal{C}} \|\Gamma(x) - \Gamma(x^*)\| \left(\frac{S^* - S_\delta}{S^* - S_\delta/2}\right)^{k-\bar{\mathcal{N}}}, \quad \forall\, k \geq \bar{\mathcal{N}} \\
&= \Big(1 + \sup_{x\in A_\delta\cap\mathcal{C}} \|\Gamma(x) - \Gamma(x^*)\|\Big)\zeta, \quad \forall\, k \geq \widehat{\mathcal{N}},
\end{aligned}$$

where $\widehat{\mathcal{N}}$ is given by $\widehat{\mathcal{N}} := \max\left\{\bar{\mathcal{N}},\ \left\lceil \bar{\mathcal{N}} + \ln\zeta / \ln\left(\frac{S^*-S_\delta}{S^*-S_\delta/2}\right)\right\rceil\right\}$.

And since ζ is arbitrary, we have

$$\lim_{k\to\infty} E_{g_k}[\Gamma(X)] = \Gamma(x^*).$$

The proof is completed by applying Lemma 5.3.2 to both **Case** 1 and **Case** 2. ∎

Remark 5.3.2 *Note that for problems with finite solution spaces, assumptions* A1 *and* A2 *are automatically satisfied. Furthermore, if we take the input parameter* $\varepsilon = 0$, *then step* 2 *of* MRAS$_0$ *is equivalent to* $\bar{\gamma}_{k+1} = \max_{1\leq i\leq k+1}\gamma_i$. *Thus,* $\{\bar{\gamma}_k\}$ *is non-decreasing and each strict increment in the sequence is bounded from below by*

$$\min_{\substack{H(x)\neq H(y)\\ x,y\in\mathcal{X}}} |H(x) - H(y)|.$$

Therefore, the $\varepsilon > 0$ *assumption in Theorem 5.3.1 can be relaxed to* $\varepsilon \geq 0$.

We now address some of the special cases discussed in Remark 5.3.1.

Corollary 5.3.2 (Multivariate Normal) *For continuous optimization problems in $\Re^n$, if multivariate normal p.d.f.'s are used in MRAS$_0$, i.e.,*

$$f(x,\theta_k) = \frac{1}{\sqrt{(2\pi)^n|\Sigma_k|}} \exp\left(-\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)\right), \tag{5.15}$$

where $\theta_k := (\mu_k;\ \Sigma_k)$, $\varepsilon > 0$, and assumptions A1−A4 are satisfied, then

$$\lim_{k\to\infty} \mu_k = x^*, \quad \textit{and} \quad \lim_{k\to\infty} \Sigma_k = 0_{n\times n},$$

where $0_{n\times n}$ represents an n-by-n zero matrix.

Proof: By Lemma 5.3.2, it is easy to show that

$$\mu_{k+1} = E_{g_{k+1}}(X), \quad \forall\, k = 0,1,\ldots,$$

and

$$\Sigma_{k+1} = E_{g_{k+1}}\left[(X-\mu_{k+1})(X-\mu_{k+1})^T\right], \quad \forall\, k = 0,1,\ldots.$$

The rest of the proof amounts to showing that

$$\lim_{k\to\infty} E_{g_k}(X) = x^*, \text{ and } \lim_{k\to\infty} E_{g_k}\left[(X-\mu_k)(X-\mu_k)^T\right] = 0_{n\times n},$$

which is the same as the proof of Theorem 5.3.1. ∎

Remark 5.3.3 *Corollary 5.3.2 shows that in the multivariate normal case, the sequence of parameterized p.d.f.'s will converge to a degenerate p.d.f. concentrated only on the optimal solution. In this case the parameters are updated as*

$$\mu_{k+1} = \frac{E_{\theta_k}\left[\{[S(H(X))]^k/f(X,\theta_k)\}I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}X\right]}{E_{\theta_k}\left[\{[S(H(X))]^k/f(X,\theta_k)\}I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]}, \tag{5.16}$$

and

$$\Sigma_{k+1} = \frac{E_{\theta_k}\left[\{[S(H(X))]^k/f(X,\theta_k)\}I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}(X-\mu_{k+1})(X-\mu_{k+1})^T\right]}{E_{\theta_k}\left[\{[S(H(X))]^k/f(X,\theta_k)\}I_{\{H(X)\geq\bar{\gamma}_{k+1}\}}\right]}, \tag{5.17}$$

where $f(x, \theta_k)$ is given by (5.15). Note that when the solution space $\mathcal{X}$ is a (simple) constrained region in $\Re^n$, one straightforward approach is to use the acceptance-rejection method (cf. e.g., [51]). And it is easy to verify that the parameter updating rules remain the same.

Corollary 5.3.3 (Independent Univariate) *If the components of the random vector $X = (X_1, \ldots, X_n)$ are independent, each has a univariate p.d.f./p.m.f. of the form*

$$f(x_i, \vartheta_i) = \exp(x_i \vartheta_i - K(\vartheta_i))h(x_i), \ \vartheta_i \in \Re, \forall\, i = 1, \ldots, n,$$

$\varepsilon > 0$, and A1−A5 are satisfied, then

$$\lim_{k \to \infty} E_{\theta_k}[X] = x^*, \quad \textit{where} \ \ \theta_k := (\vartheta_1^k, \ldots, \vartheta_n^k).$$

5.4 An Alternative View of the Cross-Entropy Method

In this Chapter, we give an alternative interpretation of the CE method for optimization and discuss its similarities and differences with the MRAS$_0$ algorithm. Specifically, we show that the CE method can also be viewed as a search strategy guided by a sequence of reference models. From this particular point of view, we establish some important properties of the CE method.

The deterministic version of the CE method for solving (5.1) can be summarized as follows.

Algorithm CE$_0$: Deterministic Version of the CE Method

1. *Choose the initial p.d.f./p.m.f. $f(\cdot, \theta_0)$, $\theta_0 \in \Theta$. Specify the parameter $\rho \in (0, 1]$ and a non-decreasing function $\varphi(\cdot) : \Re \to \Re^+ \cup \{0\}$. Set $k = 0$.*

2. *Calculate the* $(1-\rho)$*-quantile* γ_{k+1} *as*

$$\gamma_{k+1} := \sup\left\{l : P_{\theta_k}(H(X) \geq l) \geq \rho\right\}.$$

3. *Compute the new parameter*

$$\theta_{k+1} := \arg\max_{\theta\in\Theta} E_{\theta_k}\left[\varphi(H(X)) I_{\{H(X)\geq\gamma_{k+1}\}} \ln f(X,\theta)\right].$$

4. *If a specified stopping rule is satisfied, then terminate; otherwise set* $k = k+1$ *and go to Step* 2.

In CE$_0$, choosing $\varphi(H(x)) = 1$ gives the standard CE method, whereas choosing $\varphi(H(x)) = H(x)$ (if $H(x) \geq 0,\ \forall\, x \in \mathcal{X}$) gives an extended version of the standard CE method (cf. e.g., [26]).

One resemblance between CE and MRAS$_0$ is the use of the parameter ρ and the $(1-\rho)$-quantile in both algorithms. However, the fundamental difference is that in CE, the problem of estimating the optimal value of the parameter is broken down into a sequence of simple estimation problems, in which the parameter ρ assumes a crucial role. Since a small change in the values of ρ may disturb the whole estimation process and affect the quality of the resulting estimates, the convergence of CE cannot be always guaranteed unless the value of ρ is chosen sufficiently small (cf. [26], [41]; also Example 5.4.1 below), whereas the theoretical convergence of MRAS$_0$ is unaffected by the parameter ρ.

The following lemma provides a unified view of MRAS and CE; it shows that by appropriately defining a sequence of implicit reference models $\{g_k^{ce}(\cdot) : k = 1, 2, \ldots\}$, the CE method can be recovered, and the parameter updating in CE is guided by this sequence of models.

Lemma 5.4.1 *The parameter θ_{k+1} computed at the kth iteration of the CE$_0$ algorithm minimizes the KL-divergence $\mathcal{D}\left(g^{ce}_{k+1}, f(\cdot,\theta)\right)$, where*

$$g^{ce}_{k+1}(x) := \frac{\varphi(H(x))I_{\{H(x)\geq\gamma_{k+1}\}}f(x,\theta_k)}{E_{\theta_k}\left[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}\right]} \quad \forall\, x \in \mathcal{X}, \quad k = 0,1,\ldots. \tag{5.18}$$

Proof: Similar to the proof of Lemma 5.3.1. ∎

The key observation to note is that in contrast to MRAS$_0$, the sequence of reference models in CE depends explicitly on the family of parameterized p.d.f's/p.m.f's $\{f(\cdot,\theta_k)\}$ used. Since $g^{ce}_{k+1}(\cdot)$ is obtained by tilting $f(\cdot,\theta_k)$ with the performance function, it improves the expected performance in the sense that

$$\begin{aligned} E_{g^{ce}_{k+1}}\left[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}\right] &= \frac{E_{\theta_k}\left[(\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}})^2\right]}{E_{\theta_k}\left[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}\right]} \\ &\geq E_{\theta_k}\left[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}\right]. \end{aligned}$$

Thus, it is reasonable to expect that the projection of $g^{ce}_{k+1}(\cdot)$ on $\{f(\cdot,\theta) : \theta \in \Theta\}$ (i.e., $f(\cdot,\theta_{k+1})$) also improves the expected performance. This result is formalized in the following theorem.

Theorem 5.4.1 *For the CE$_0$ algorithm, we have*

$$E_{\theta_{k+1}}\left[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}\right] \geq E_{\theta_k}\left[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}\right], \quad \forall\, k = 0,1,\ldots.$$

Proof: Define $\widehat{g}^{ce}_{k+2}(\cdot)$ as

$$\widehat{g}^{ce}_{k+2}(x) := \frac{\varphi(H(x))I_{\{H(x)\geq\gamma_{k+1}\}}f(x,\theta_{k+1})}{E_{\theta_{k+1}}\left[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}\right]} \quad \forall\, x \in \mathcal{X},\ k = 0,1,\ldots.$$

We have from the definition of $g^{ce}_{k+1}(\cdot)$,

$$\begin{aligned} \mathcal{D}(g^{ce}_{k+1}, \widehat{g}^{ce}_{k+2}) &= E_{g^{ce}_{k+1}}\left[\ln\frac{g^{ce}_{k+1}(X)}{\widehat{g}^{ce}_{k+2}(X)}\right] \\ &= E_{g^{ce}_{k+1}}\left[\ln\frac{f(X,\theta_k)}{f(X,\theta_{k+1})}\right] + \ln\frac{E_{\theta_{k+1}}[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}]}{E_{\theta_k}[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}]}. \end{aligned}$$

Since θ_{k+1} minimizes the K-L divergence $\mathcal{D}(g^{ce}_{k+1}, f(\cdot,\theta))$ (cf. Lemma 5.4.1), it follows that

$$\begin{aligned}
0 &\leq \mathcal{D}(g^{ce}_{k+1}, f(\cdot,\theta_k)) - \mathcal{D}(g^{ce}_{k+1}, f(\cdot,\theta_{k+1})) \\
&\leq \mathcal{D}(g^{ce}_{k+1}, f(\cdot,\theta_k)) - \mathcal{D}(g^{ce}_{k+1}, f(\cdot,\theta_{k+1})) + \mathcal{D}(g^{ce}_{k+1}, \widehat{g}^{ce}_{k+2}) \\
&= E_{g^{ce}_{k+1}}\left[\ln \frac{f(X,\theta_{k+1})}{f(X,\theta_k)}\right] + E_{g^{ce}_{k+1}}\left[\ln \frac{f(X,\theta_k)}{f(X,\theta_{k+1})}\right] + \ln \frac{E_{\theta_{k+1}}[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}]}{E_{\theta_k}[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}]} \\
&= \ln \frac{E_{\theta_{k+1}}[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}]}{E_{\theta_k}[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}]}
\end{aligned}$$

Therefore

$$E_{\theta_{k+1}}[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}] \geq E_{\theta_k}[\varphi(H(X))I_{\{H(X)\geq\gamma_{k+1}\}}].$$

∎

In the standard CE method, Theorem 5.4.1 implies the monotonicity of the sequence $\{\gamma_k : k = 1, 2, \ldots\}$.

Lemma 5.4.2 *For the standard CE method (i.e., CE$_0$ with $\varphi(H(x)) = 1$), we have*

$$\gamma_{k+2} \geq \gamma_{k+1}, \quad \forall\, k = 0, 1, \ldots.$$

Proof: By Theorem 5.4.1, we have

$$E_{\theta_{k+1}}[I_{\{H(X)\geq\gamma_{k+1}\}}] \geq E_{\theta_k}[I_{\{H(X)\geq\gamma_{k+1}\}}],$$

i.e.,

$$P_{\theta_{k+1}}(H(X) \geq \gamma_{k+1}) \geq P_{\theta_k}(H(X) \geq \gamma_{k+1}) \geq \rho.$$

The result follows by the definition of γ_{k+2} (See Step 2 of the CE$_0$ algorithm). ∎

Note that since $\gamma_k \leq H(x^*)$ for all k, Lemma 5.4.2 implies that the sequence $\{\gamma_k : k = 1, \ldots\}$ generated by the standard CE method converges. However, depending on the p.d.f's/p.m.f's and the parameter ρ used, the sequence $\{\gamma_k\}$ may not converge to $H(x^*)$ or even to a small neighborhood of $H(x^*)$ (cf. Examples 4.1 and 4.2 below).

Similar to MRAS$_0$ (cf. Lemma 5.3.2), when $f(\cdot,\theta)$ belongs to the natural exponential families, the following lemma relates the sequence $\{f(\cdot,\theta_k),\ k=1,2,\ldots\}$ to the sequence of reference models $\{g_k^{ce}(\cdot) : k=1,2,\ldots\}$.

Lemma 5.4.3 *Assume that:*

1. *There exists a compact set $\bar{\Pi}$ such that the level set $\{x : H(x) \geq \gamma_k\} \cap \mathcal{X} \subseteq \bar{\Pi}$ for all $k=1,2,\ldots$, where $\gamma_k = \sup_l\{l : P_{\theta_{k-1}}(H(X) \geq l) \geq \rho\}$ is defined as in the CE$_0$ algorithm.*

2. *The parameter θ_{k+1} computed at step 3 of the CE$_0$ algorithm is an interior point of Θ for all k.*

3. *Assumptions A5 is satisfied.*

Then

$$E_{\theta_{k+1}}[\Gamma(X)] = E_{g_{k+1}^{ce}}[\Gamma(X)], \quad \forall\, k=0,1,\ldots.$$

The above lemma indicates that the behavior of the sequence of p.d.f's/p.m.f's $\{f(\cdot,\theta_k)\}$ is closely related to the properties of the sequence of reference models. To understand this, consider the particular case where $\Gamma(x) = x$. If the CE method converges to the optimal solution in the sense that $\lim_{k\to\infty} E_{\theta_k}[H(X)] = H(x^*)$, then we must have $\lim_{k\to\infty} E_{\theta_k}[X] = x^*$, since $H(x) < H(x^*)\ \forall\, x \neq x^*$. Thus, by Lemma 5.4.3, a necessary condition for this convergence is $\lim_{k\to\infty} E_{g_k^{ce}}[X] = x^*$. However, unlike MRAS$_0$, where the convergence of the sequence of reference models to an optimal degenerate distribution is guaranteed, the convergence of the sequence $\{g_k^{ce}(\cdot) : k=1,2,\ldots\}$ relies on the choices of the families of distributions $\{f(\cdot,\theta)\}$ and the values of the parameter ρ used (cf. (5.18)). We now illustrate this issue by two simple examples.

Example 5.4.1 **(The Standard CE Method)** Consider maximizing the function $H(x)$ given by

$$H(x) = \begin{cases} 0 & x \in \{(0,1),(1,0)\}, \\ 1 & x = (0,0), \\ a & x = (1,1), \end{cases} \tag{5.19}$$

where $a > 1$, and $x := (x_1, x_2) \in \mathcal{X} := \{(0,0),(0,1),(1,0),(1,1)\}$.

If we take $0.25 < \rho \leq 0.5$ and an initial p.m.f.

$$f(x,\theta_0) = {p_0}^{x_1}(1-p_0)^{1-x_1}{q_0}^{x_2}(1-q_0)^{1-x_2} \text{ with } \theta_0 = (p_0,q_0) = (0.5,0.5),$$

then since $P_{\theta_0}(x \in \{(0,0),(1,1)\}) = 0.5 \geq \rho$, we have $\gamma_1 = 1$. It is also straightforward to see that

$$g_1^{ce}(x) = \begin{cases} 0.5 & x = (0,0) \text{ or } (1,1), \\ 0 & \text{otherwise}, \end{cases}$$

and the parameter θ_1 computed at step 3 (with $\varphi(H(x)) = 1$) of CE$_0$ is given by $\theta_1 = (0.5, 0.5)$. Proceeding iteratively, we have $\gamma_k = 1$ and $g_k^{ce}(x) = g_1^{ce}(x)\ \forall\, k = 1,2,\ldots$, i.e., the algorithm does not converge to a degenerate distribution at the optimal solution.

On the other hand, if we choose $\rho \leq 0.25$, then it turns out that $\gamma_k = a$ and

$$g_k^{ce}(x) = \begin{cases} 1 & x = (1,1), \\ 0 & \text{otherwise}. \end{cases}$$

for all $k = 1,2,\ldots$, which means the algorithm converges to the optimum.

Example 5.4.2 **(The Extended Version of the CE Method)** Consider solving problem (5.19) by CE$_0$ with the performance function $\varphi(H(x)) = H(x)$. We use the same family of p.m.f's as in Example 5.4.1 with the initial parameter $\theta_0 = (\frac{1}{1+a}, \frac{1}{1+a})$. If the values of ρ are chosen from the interval $\left(\frac{1}{(1+a)^2}, \frac{a^2+1}{(1+a)^2}\right)$, then we have $\theta_k = (\frac{1}{1+a}, \frac{1}{1+a})$, $\gamma_k = 1$, and

$$g_k^{ce}(x) = \begin{cases} \frac{a}{1+a} & x = (0,0), \\ \frac{1}{1+a} & x = (1,1), \\ 0 & \text{otherwise}, \end{cases}$$

for all $k = 1, 2, \ldots$.

On the other hand, if we choose $\rho = 0.5$ and $\theta_0 = (0.5, 0.5)$, then it is easy to verify that $\lim_{k\to\infty} \gamma_k = a$ and

$$\lim_{k\to\infty} g_k^{ce}(x) = \begin{cases} 1 & x = (1,1), \\ 0 & \text{otherwise}. \end{cases}$$

5.5 The MRAS$_1$ Algorithm (Monte Carlo Version)

The MRAS$_0$ algorithm describes the idealized situation where quantile values and expectations can be evaluated exactly. In practice, we will usually resort to its stochastic counterpart, where only a finite number of samples are used and expected values are replaced with their corresponding sample averages. For example, step 3 of MRAS$_0$ will be replaced with

$$\widetilde{\theta}_{k+1} = \arg\max_{\theta\in\Theta} \frac{1}{N} \sum_{i=1}^{N} \frac{[S(H(X_i))]^k}{f(X_i, \widetilde{\theta}_k)} I_{\{H(X_i)\geq\bar{\gamma}_{k+1}\}} \ln f(X_i, \theta), \tag{5.20}$$

where $X_1, \ldots, X_N$ are i.i.d. random samples generated from $f(x, \widetilde{\theta}_k)$, $\widetilde{\theta}_k$ is the estimated parameter vector computed at the previous iteration, and $\bar{\gamma}_{k+1}$ is a threshold determined by the sample $(1-\rho)$-quantile of $H(X_1), \ldots, H(X_N)$.

However, the theoretical convergence can no longer be guaranteed for a simple stochastic counterpart of MRAS$_0$. In particular, the set $\{x : H(x) \geq \bar{\gamma}_{k+1}\}$ involved in (5.20) may be empty, since all the random samples generated at the current iteration may be much worse than those generated at the previous iteration. Thus, we can only expect the algorithm to converge if the expected values in the MRAS$_0$ algorithm are closely approximated. Obviously, the quality of the approximation will depend on the number of samples to be used in the simulation, but it is difficult to determine in advance the appropriate number of samples. A sample size too small will cause the algorithm to fail to converge and result in poor quality solutions, whereas a sample size too large may lead to high computational cost.

As mentioned earlier, the parameter ρ, to some extent, will affect the performance of the algorithm. Large values of ρ mean that almost all samples generated, regardless of their performances, will be used to update the probabilistic model, which could slow down the convergence process. On the other hand, since a good estimate will necessarily require a reasonable amount of valid samples, the quantity ρN (i.e., the approximate amount of samples that will be used in parameter updating) cannot be too small. Thus, small values of ρ will require a large number of samples to be generated at each iteration and may result in significant simulation efforts. For a given problem, although it is clear that we should avoid those values of ρ that are either too close to 1 or too close to 0, to determine a priori which ρ gives a satisfactory performance may be difficult.

In order to address the above difficulties, we adopt the same idea as in [41] and propose a modified Monte Carlo version of MRAS$_0$ in which the sample size N is adaptively increasing and the parameter ρ is adaptively decreasing.

5.5.1 Algorithm Description

Roughly speaking, the MRAS$_1$ algorithm is essentially a Monte Carlo version of MRAS$_0$ except that the parameter ρ and the sample size N may change from one iteration

Algorithm MRAS$_1$ – Monte Carlo version

```
• Initialization: Specify $\rho_0 \in (0,1]$, an initial sample size $N_0 > 1$, $\varepsilon \geq 0$, $\alpha > 1$, a mixing coefficient $\lambda \in (0,1]$, a strictly increasing function $S(\cdot) : \Re \to \Re^+$, and an initial p.d.f. $f(x,\theta_0) > 0 \; \forall x \in \mathcal{X}$. Set $\widetilde{\theta}_0 \leftarrow \theta_0$, $k \leftarrow 0$.

• Repeat until a specified stopping rule is satisfied:

    1. Generate $N_k$ i.i.d. samples $X_1^k, \ldots, X_{N_k}^k$ according to $\widetilde{f}(\cdot,\widetilde{\theta}_k) := (1-\lambda) f(\cdot,\widetilde{\theta}_k) + \lambda f(\cdot,\theta_0)$.

    2. Compute the sample $(1-\rho_k)$-quantile $\widetilde{\gamma}_{k+1}(\rho_k, N_k) := H_{(\lceil (1-\rho_k)N_k \rceil)}$, where $\lceil a \rceil$ is the smallest integer greater than $a$, and $H_{(i)}$ is the $i$th order statistic of the sequence $\{H(X_i^k),\ i = 1,\ldots,N_k\}$.

    3. If $k = 0$ or $\widetilde{\gamma}_{k+1}(\rho_k, N_k) \geq \bar{\gamma}_k + \frac{\varepsilon}{2}$, then

            3a. Set $\bar{\gamma}_{k+1} \leftarrow \widetilde{\gamma}_{k+1}(\rho_k, N_k),\ \rho_{k+1} \leftarrow \rho_k,\ N_{k+1} \leftarrow N_k$.

        else, find the largest $\bar{\rho} \in (0,\rho_k)$ such that $\widetilde{\gamma}_{k+1}(\bar{\rho}, N_k) \geq \bar{\gamma}_k + \frac{\varepsilon}{2}$.

            3b. If such a $\bar{\rho}$ exists, then set $\bar{\gamma}_{k+1} \leftarrow \widetilde{\gamma}_{k+1}(\bar{\rho}, N_k),\ \rho_{k+1} \leftarrow \bar{\rho},\ N_{k+1} \leftarrow N_k$.

            3c. else (if no such $\bar{\rho}$ exists), set $\bar{\gamma}_{k+1} \leftarrow \bar{\gamma}_k,\ \rho_{k+1} \leftarrow \rho_k,\ N_{k+1} \leftarrow \lceil \alpha N_k \rceil$.

        endif

    4. Compute $\widetilde{\theta}_{k+1}$ as

        $\widetilde{\theta}_{k+1} = \arg\max_{\theta \in \Theta} \frac{1}{N_k} \sum_{i=1}^{N_k} \frac{[S(H(X_i^k))]^k}{\widetilde{f}(X_i^k, \widetilde{\theta}_k)} I_{\{H(X_i^k) \geq \bar{\gamma}_{k+1}\}} \ln f(X_i^k, \theta).$    (5.21)

    5. Set $k \leftarrow k+1$.
```

to another. The rate of increase in the sample size is controlled by an extra parameter $\alpha > 1$, specified during the initialization step. For example, if the initial sample size is N_0, then after k increments, the sample size will be approximately $\lceil \alpha^k N_0 \rceil$.

At each iteration k, random samples are drawn from the density/mass function $\widetilde{f}(\cdot,\widetilde{\theta}_k)$, which is a mixture of the initial density/mass $f(\cdot,\theta_0)$ and the density/mass calculated from the previous iteration $f(\cdot,\widetilde{\theta}_k)$ (cf. e.g., [9] for a similar idea in the context of multiarmed bandit models). We assume that $f(\cdot,\theta_0)$ satisfies the following condition:

Assumption A3′. *There exists a compact set* Π_ε *such that* $\{x : H(x) \geq H(x^*) - \varepsilon\} \cap \mathcal{X} \subseteq \Pi_\varepsilon$*. Moreover, the initial density/mass function* $f(x,\theta_0)$ *is bounded away from zero on* Π_ε*, i.e.,* $f_* := \inf_{x\in\Pi_\varepsilon} f(x,\theta_0) > 0$.

In practice, the initial density $f(\cdot,\theta_0)$ can be chosen according to some prior knowledge of the problem structure; however, if nothing is known about where the good solutions are, this density should be chosen in such a way that each region in the solution space will have an (approximately) equal probability of being sampled. For instance, when $\mathcal{X}$ is finite, one simple choice of $f(\cdot,\theta_0)$ is the uniform distribution. Intuitively, mixing in the initial density forces the algorithm to explore the entire solution space and to maintain a global perspective during the search process. Also note that if $\lambda = 1$, then random samples will always be drawn from the initial density, in which case, MRAS$_1$ becomes a pure random sampling approach.

At step 2, the sample $(1-\rho_k)$-quantile $\widetilde{\gamma}_{k+1}$ is calculated by first ordering the sample performances $H(X_i^k)$, $i = 1,\dots,N_k$ from smallest to largest, $H_{(1)} \leq H_{(2)} \leq \cdots \leq H_{(N_k)}$, and then taking the $\lceil(1-\rho_k)N_k\rceil$th order statistic. We use the function $\widetilde{\gamma}_{k+1}(\rho_k, N_k)$ to emphasize the dependencies of $\widetilde{\gamma}_{k+1}$ on both ρ_k and N_k, so that different sample quantile values used during one iteration can be distinguished by their arguments.

Step 3 of MRAS$_1$ is used to extract a sequence of non-decreasing thresholds $\{\bar{\gamma}_k, k = 1, 2\dots\}$ from the sequence of sample quantiles $\{\widetilde{\gamma}_k\}$, and to determine the appropriate values of ρ_{k+1} and N_{k+1} to be used in subsequent iterations. This step is carried out as follows. At each iteration k, we first check whether the inequality $\widetilde{\gamma}_{k+1}(\rho_k, N_k) \geq \bar{\gamma}_k + \varepsilon/2$ is satisfied, where $\bar{\gamma}_k$ is the threshold value used in the previous iteration. If the inequality holds, then it means that both the current ρ_k value and the current sample size N_k are satisfactory; thus we proceed to step $3a$ and update the parameter vector $\widetilde{\theta}_{k+1}$ in step 4 by using $\widetilde{\gamma}_{k+1}(\rho_k, N_k)$. Otherwise, it indicates that either ρ_k is too large or the sample size N_k is too small. To determine which, we fix the sample size N_k and check if there exists a smaller $\bar{\rho} < \rho_k$ such that the above inequality can be satisfied with the new sample $(1-\bar{\rho})$-quantile. If such a $\bar{\rho}$ does exist, then the current sample size N_k is still deemed acceptable, and we only need to decrease the ρ_k value. Accordingly, the parameter vector is updated in step 4 by using the sample $(1-\bar{\rho})$-quantile. On the other hand, if no such $\bar{\rho}$ can be found, then the parameter vector is updated by using the threshold $\bar{\gamma}_k$ calculated during the previous iteration and the sample size N_k is increased by a factor α.

We make the following assumption about the parameter vector $\widetilde{\theta}_{k+1}$ computed at step 4:

Assumption A4′. The parameter vector $\widetilde{\theta}_{k+1}$ computed at step 4 of MRAS$_1$ is an interior point of Θ for all k.

It is important to note that the set $\{x : H(x) \geq \bar{\gamma}_{k+1}, x \in \{X_1^k, \ldots, X_{N_k}^k\}\}$ could be empty if step $3c$ is visited. If this happens, the right hand side of (5.21) will be equal to zero, so any $\theta \in \Theta$ is a maximizer, and we define $\widetilde{\theta}_{k+1} := \widetilde{\theta}_k$ in this case.

5.5.2 Global Convergence

In this Chapter, we discuss the convergence properties of the MRAS$_1$ algorithm for natural exponential families (NEFs). To be specific, we will explore the relations between

MRAS$_1$ and MRAS$_0$ and show that with high probability, the gaps (e.g., approximation errors incurred by replacing expected values with sample averages) between the two algorithms can be made small enough such that the convergence analysis of MRAS$_1$ can be ascribed to the convergence analysis of the MRAS$_0$ algorithm; thus, our analysis relies heavily on the results obtained in Chapter 5.3.2. Throughout this Chapter, we denote by $P_{\widetilde{\theta}_k}(\cdot)$ and $E_{\widetilde{\theta}_k}[\cdot]$ the respective probability and expectation taken with respect to the p.d.f./p.m.f. $f(\cdot,\widetilde{\theta}_k)$, and $\widetilde{P}_{\widetilde{\theta}_k}(\cdot)$ and $\widetilde{E}_{\widetilde{\theta}_k}[\cdot]$ the respective probability and expectation taken with respect to $\widetilde{f}(\cdot,\widetilde{\theta}_k)$. Note that since the sequence $\{\widetilde{\theta}_k\}$ results from random samples generated at each iteration of MRAS$_1$, these quantities are also random.

Let $\widetilde{g}_{k+1}(\cdot),\ k=0,1,\ldots,$ be defined by

$$\widetilde{g}_{k+1}(x) := \begin{cases} \frac{\left[[S(H(x))]^k/\widetilde{f}(x,\widetilde{\theta}_k)\right] I_{\{H(x)\geq\bar{\gamma}_{k+1}\}}}{\sum_{i=1}^{N_k}\left[[S(H(X_i^k))]^k/\widetilde{f}(X_i^k,\widetilde{\theta}_k)\right] I_{\{H(X_i^k)\geq\bar{\gamma}_{k+1}\}}} & \text{if } \{x : H(x)\geq\bar{\gamma}_{k+1}, x\in\Lambda_k\}\neq\emptyset, \\ \widetilde{g}_k(x) & \text{otherwise,} \end{cases} \tag{5.22}$$

where $\Lambda_k := \left\{X_1^k,\ldots,X_{N_k}^k\right\}$ is the population of candidate solutions generated at iteration k, and $\bar{\gamma}_{k+1}$ is given by $\bar{\gamma}_{k+1} := \begin{cases} \widetilde{\gamma}_{k+1}(\rho_k, N_k) & \text{if step } 3a \text{ is visited,} \\ \widetilde{\gamma}_{k+1}(\bar{\rho}, N_k) & \text{if step } 3b \text{ is visited,} \\ \bar{\gamma}_k & \text{if step } 3c \text{ is visited.} \end{cases}$

Similar to Lemma 5.3.2, the following lemma shows the connection between $f(\cdot,\widetilde{\theta}_{k+1})$ and $\widetilde{g}_{k+1}(\cdot)$.

Lemma 5.5.1 *If assumptions $A4'$ and $A5$ hold, then the parameter $\widetilde{\theta}_{k+1}$ computed at step 3 of MRAS$_1$ satisfies*

$$E_{\widetilde{\theta}_{k+1}}\left[\Gamma(X)\right] = E_{\widetilde{g}_{k+1}}\left[\Gamma(X)\right], \quad \forall\, k=0,1,\ldots,$$

Note that the region $\{x : H(x)\geq\bar{\gamma}_{k+1}\}$ will become smaller and smaller as $\bar{\gamma}_{k+1}$

increases. Lemma 5.5.1 shows that the sequence of sampling p.d.f's/p.m.f's $\{f(\cdot,\widetilde{\theta}_{k+1})\}$ is adapted to this sequence of shrinking regions. For example, consider the case where $\{x : H(x) \geq \bar{\gamma}_{k+1}\}$ is convex and $\Gamma(x) = x$. Since $E_{\widetilde{g}_{k+1}}[X]$ is the convex combination of $X_1^k, \dots, X_{N_k}^k$, the lemma implies that $E_{\widetilde{\theta}_{k+1}}[X] \in \{x : H(x) \geq \bar{\gamma}_{k+1}\}$. Thus, it is natural to expect that the random samples generated at the next iteration will fall in the region $\{x : H(x) \geq \bar{\gamma}_{k+1}\}$ with large probabilities (e.g., consider the normal p.d.f. where its mode is equal to its mean). In contrast, if we use a fixed sampling distribution for all iterations as in pure random sampling (i.e., the $\lambda = 1$ case), then sampling from this sequence of shrinking regions could become a substantially difficult problem in practice.

Next, we present a useful lemma, which shows the convergence of the quantile estimates when random samples are generated from a sequence of different distributions.

Lemma 5.5.2 *For any given $\rho^\dagger \in (0,1)$, let $\gamma_k^\dagger$ be the set of $(1-\rho^\dagger)$-quantiles of $H(X)$ with respect to the p.d.f./p.m.f. $\widetilde{f}(\cdot,\widetilde{\theta}_k)$, and let $\widetilde{\gamma}_k^\dagger(\rho^\dagger, N_k)$ be the corresponding sample quantile of $H(X_1^k), \dots, H(X_{N_k}^k)$, where $\widetilde{f}(\cdot,\widetilde{\theta}_k)$ and N_k are defined as in MRAS$_1$, and $X_1^k, \dots, X_{N_k}^k$ are i.i.d. with common density $\widetilde{f}(\cdot,\widetilde{\theta}_k)$. Then the distance from $\widetilde{\gamma}_k^\dagger(\rho^\dagger, N_k)$ to $\gamma_k^\dagger$ tends to zero as $k \to \infty$ w.p.1.*

Proof: Our proof is based on the proof of Lemma $A1$ in [69]. Notice that for given $\rho^\dagger$ and $\widetilde{f}(\cdot,\widetilde{\theta}_k)$, $\gamma_k^\dagger$ can be obtained as the optimal solution of the following problem (cf. [41])

$$\min_{v \in \mathcal{V}} \ell_k(v), \tag{5.23}$$

where $\mathcal{V} = [0, H(x^*)]$, $\ell_k(v) := \widetilde{E}_{\widetilde{\theta}_k} \phi(H(X), v)$, and

$$\phi(H(x), v) := \begin{cases} (1-\rho^\dagger)(H(x) - v) & \text{if } v \leq H(x), \\ \rho^\dagger(v - H(x)) & \text{if } v \geq H(x). \end{cases}$$

Similarly, the sample quantile $\widetilde{\gamma}_k^\dagger(\rho^\dagger, N_k)$ can be expressed as the solution to the sample average approximation of (5.23),

$$\min_{v\in\mathcal{V}} \bar{\ell}_k(v), \tag{5.24}$$

where $\bar{\ell}_k(v) := \frac{1}{N_k}\sum_{j=1}^{N_k}\phi(H(X_j^k), v)$ and $X_1^k, \ldots, X_{N_k}^k$ are i.i.d. with density $\widetilde{f}(\cdot, \widetilde{\theta}_k)$.

Since the function $\phi(H(x), v)$ is bounded and continuous on $\mathcal{V}$ for all $x \in \mathcal{X}$, it is not difficult to show that $\ell_k(v)$ is continuous on $\mathcal{V}$ (cf. [69]).

Now consider a point $v \in \mathcal{V}$ and let $B_i \subseteq \mathcal{V}$ be a sequence of open balls containing v such that $B_{i+1} \subseteq B_i \;\forall\, i$ and $\lim_{L\to\infty} \cap_{i=1}^L B_i = v$. Define the function

$$b_i(H(x)) := \sup\left\{|\phi(H(x), u) - \phi(H(x), v)| : \; u \in B_i\right\}.$$

We have from the dominated convergence theorem

$$\lim_{i\to\infty} \widetilde{E}_{\widetilde{\theta}_k}[b_i(H(X))] = \widetilde{E}_{\widetilde{\theta}_k}[\lim_{i\to\infty} b_i(H(X))] = 0 \quad \forall\, k = 1, 2, \ldots, \tag{5.25}$$

where the last equality follows from the fact that $\phi(H(x), v)$ is continuous on $\mathcal{V}$.

Since

$$|\bar{\ell}_k(u) - \bar{\ell}_k(v)| \le \frac{1}{N_k}\sum_{j=1}^{N_k} |\phi(H(X_j^k), u) - \phi(H(X_j^k), v)|,$$

it follows that

$$\sup_{u\in B_i} |\bar{\ell}_k(u) - \bar{\ell}_k(v)| \le \frac{1}{N_k}\sum_{j=1}^{N_k} b_i(H(X_j^k)). \tag{5.26}$$

We now show that $\frac{1}{N_k}\sum_{j=1}^{N_k} b_i(H(X_j^k)) \to \widetilde{E}_{\widetilde{\theta}_k}[b_i(H(X))]$ as $k \to \infty$ w.p.1.

Let M be an upperbound for $b_i(H(x))$, and let $\mathcal{T}_\varepsilon := \lceil \frac{2[H(x^*)-\mathcal{M}]}{\varepsilon} \rceil$, where $\mathcal{M}$ is a lower bound for the function $H(x)$, and ε is defined as in the MRAS$_1$ algorithm. Note that the total number of visits to step $3a$ and $3b$ of MRAS$_1$ is bounded by $\mathcal{T}_\varepsilon$, thus for any $k > \mathcal{T}_\varepsilon$, the total number of visits to step $3c$ is greater than $k - \mathcal{T}_\varepsilon$. Since conditional on $\widetilde{\theta}_k$, $\frac{1}{N_k}\sum_{j=1}^{N_k} b_i(H(X_j^k))$ is an unbiased estimate of $\widetilde{E}_{\widetilde{\theta}_k}[b_i(H(X))]$, by the Hoeffding inequality

([40]), for any $\zeta > 0$,

$$P\Big(\big| \frac{1}{N_k} \sum_{j=1}^{N_k} b_i(H(X_j^k)) - \widetilde{E}_{\widetilde{\theta}_k}[b_i(H(X))] \big| > \zeta \ \Big| \widetilde{\theta}_k = \theta \Big) \le 2\exp\left(\frac{-2N_k\zeta^2}{M^2} \right) \ \ \forall\, k.$$

Therefore,

$$\begin{aligned} P\Big(\big| \frac{1}{N_k} \sum_{j=1}^{N_k} b_i(H(X_j^k)) - \widetilde{E}_{\widetilde{\theta}_k}[b_i(H(X))] \big| > \zeta \Big) \quad &\le \quad 2\exp\left(\frac{-2N_k\zeta^2}{M^2} \right) \ \ \forall\, k, \\ &\le \quad 2\exp\left(\frac{-2\alpha^{k-\mathcal{T}_\varepsilon} N_0 \zeta^2}{M^2} \right) \ \ \forall\, k > \mathcal{T}_\varepsilon, \\ &\longrightarrow \quad 0 \quad \text{as } k \to \infty, \text{ since } \alpha > 1. \end{aligned}$$

Furthermore, it is easy to see that

$$\sum_{k=1}^{\infty} P\Big(\big| \frac{1}{N_k} \sum_{j=1}^{N_k} b_i(H(X_j^k)) - \widetilde{E}_{\widetilde{\theta}_k}[b_i(H(X))] \big| > \zeta \Big) \le 2 \sum_{k=1}^{\infty} \exp\left(\frac{-2\alpha^{k-\mathcal{T}_\varepsilon} N_0 \zeta^2}{M^2} \right) < \infty.$$

By the Borel-Cantelli lemma,

$$P\Big(\big| \frac{1}{N_k} \sum_{j=1}^{N_k} b_i(H(X_j^k)) - \widetilde{E}_{\widetilde{\theta}_k}[b_i(H(X))] \big| > \zeta \ \ i.o. \Big) = 0.$$

This implies that $\frac{1}{N_k}\sum_{j=1}^{N_k} b_i(H(X_j^k)) \to \widetilde{E}_{\widetilde{\theta}_k}[b_i(H(X))]$ as $k \to \infty$ w.p.1. Note that by using a similar argument as above, we can also show that $\bar{\ell}_k(v) \to \ell_k(v)$ w.p.1 as $k \to \infty$.

The above result together with (5.25) and (5.26) implies that for any $\delta > 0$, there exists a small neighborhood B_v of v such that

$$\sup\{|\bar{\ell}_k(u) - \bar{\ell}_k(v)| : \ u \in B_v\} < \delta \ \text{ w.p.1 } \ \text{for } k \text{ sufficiently large.}$$

Since this holds for all $v \in \mathcal{V}$, we have $\mathcal{V} \subseteq \cup_{v\in\mathcal{V}} B_v$, and because $\mathcal{V}$ is compact, there exists a finite subcover $B_{v_1}, \ldots, B_{v_m}$ such that

$$\sup\{|\bar{\ell}_k(u) - \bar{\ell}_k(v_j)| : \ u \in B_{v_j}\} < \delta \ \text{ w.p.1 } \ \text{for } k \text{ sufficiently large, and } \mathcal{V} \subseteq \cup_{j=1}^m B_{v_j}.$$

Furthermore, by the continuity of $\ell_k(v)$, these open balls can be chosen in such a way that

$$\sup\{|\ell_k(u) - \ell_k(v_j)| : \ u \in B_{v_j}\} < \delta \ \ \forall\, j = 1, \ldots, m.$$

Since $\bar{\ell}_k(v_j) \to \ell_k(v_j)$ w.p.1 as $k \to \infty$ for all $j = 1, \dots, m$,

$$|\bar{\ell}_k(v_j) - \ell_k(v_j)| < \delta \text{ w.p.1 for } k \text{ sufficiently large, } \forall\, j = 1, \dots, m.$$

For any $v \in \mathcal{V}$, without lost of generality assume $v \in B_{v_j}$, we have w.p.1 for k sufficiently large

$$|\bar{\ell}_k(v) - \ell_k(v)| \le |\bar{\ell}_k(v) - \bar{\ell}_k(v_j)| + |\ell_k(v) - \ell_k(v_j)| + |\bar{\ell}_k(v_j) - \ell_k(v_j)| < 3\delta,$$

which implies that $\bar{\ell}_k(v) \to \ell_k(v)$ uniformly w.p.1 on $\mathcal{V}$.

The rest of the proof follows from Theorem $A1$ in [69] (pp. 69), which basically states that if $\bar{\ell}_k(v) \to \ell_k(v)$ uniformly w.p.1, then the distance from $\widetilde{\gamma}_k^\dagger(\rho^\dagger, N_k)$ to $\gamma_k^\dagger$ tends to zero w.p.1 as $k \to \infty$. ∎

We are now ready to state the main theorem.

Theorem 5.5.1 *Let $\varepsilon > 0$, and define the ε-optimal set $\mathcal{O}_\varepsilon := \{x : H(x) \ge H(x^*) - \varepsilon\} \cap \mathcal{X}$. If assumptions $A1$, $A3'$, $A4'$, and $A5$ are satisfied, then there exists a random variable $\mathcal{K}$ such that w.p.1., $\mathcal{K} < \infty$, and*

1. $\bar{\gamma}_k > H(x^*) - \varepsilon, \quad \forall\, k \ge \mathcal{K}$
2. $E_{\widetilde{\theta}_{k+1}}[\Gamma(X)] \in CONV\{\Gamma(\mathcal{O}_\varepsilon)\}$, $\forall\, k \ge \mathcal{K}$, *where $CONV\{\Gamma(\mathcal{O}_\varepsilon)\}$ indicates the convex hull of the set $\Gamma(\mathcal{O}_\varepsilon)$.*

Furthermore, let β be a positive constant satisfying the condition that the set $\{x : S(H(x)) \ge \frac{1}{\beta}\}$ has a strictly positive Lebesgue/counting measure. If assumptions $A1$, $A2$, $A3'$, $A4'$, and $A5$ are all satisfied and $\alpha > (\beta S^*)^2$, where $S^* := S(H(x^*))$, then

3. $\lim_{k\to\infty} E_{\widetilde{\theta}_k}[\Gamma(X)] = \Gamma(x^*)$ *w.p.1.*

Remark 5.5.1 *Roughly speaking, the second result can be understood as finite time ε-optimality. To see this, consider the special case where $H(x)$ is locally concave on the set $\mathcal{O}_\varepsilon$. Let $x, y \in \mathcal{O}_\varepsilon$ and $\eta \in [0,1]$ be arbitrary. By the definition of concavity, we will have $H(\eta x + (1-\eta)y) \geq \eta H(x) + (1-\eta)H(y) \geq H(x^*) - \varepsilon$, which implies that the set $\mathcal{O}_\varepsilon$ is convex. If in addition $\Gamma(x)$ is also convex and one-to-one on $\mathcal{O}_\varepsilon$ (e.g. multivariate normal p.d.f.), then $CONV\{\Gamma(\mathcal{O}_\varepsilon)\} = \Gamma(\mathcal{O}_\varepsilon)$. Thus it follows that $\Gamma^{-1}(E_{\widetilde{\theta}_{k+1}}[\Gamma(X)]) \in \mathcal{O}_\varepsilon, \ \forall\, k \geq \mathcal{K}$ w.p.1.*

Proof of Theorem 5.5.1: **(1)** The first part of the proof is an extension of the proofs given in [41]. First we claim that given ρ_k and $\bar{\gamma}_k$, if $\bar{\gamma}_k \leq H(x^*) - \varepsilon$, then $\exists\, \bar{\mathcal{K}} < \infty$ w.p.1 and $\bar{\rho} \in (0, \rho_k)$ such that $\widetilde{\gamma}_{k'+1}(\bar{\rho}, N_{k'}) \geq \bar{\gamma}_k + \frac{\varepsilon}{2}\ \forall\, k' \geq \bar{\mathcal{K}}$. To show this, we proceed by contradiction.

Let $\rho_k^* := \widetilde{P}_{\widetilde{\theta}_k}\left(H(X) \geq \bar{\gamma}_k + \frac{2\varepsilon}{3}\right)$. If $\bar{\gamma}_k \leq H(x^*) - \varepsilon$, then $\bar{\gamma}_k + \frac{2\varepsilon}{3} \leq H(x^*) - \frac{\varepsilon}{3}$. By A1 and A3′, we have

$$\rho_k^* \geq \widetilde{P}_{\widetilde{\theta}_k}\left(H(X) \geq H(x^*) - \frac{\varepsilon}{3}\right) \geq \lambda \mathcal{C}(\varepsilon, \theta_0) > 0, \tag{5.27}$$

where $\mathcal{C}(\varepsilon, \theta_0) = \int_{\mathcal{X}} I_{\{H(x) \geq H(x^*) - \varepsilon/3\}} f(x, \theta_0)\nu(dx)$ is a constant.

Now assume that $\exists\, \rho \in (0, \rho_k^*)$ such that $\gamma_{k+1}(\rho, \widetilde{\theta}_k) < \bar{\gamma}_k + \frac{2\varepsilon}{3}$, where $\gamma_{k+1}(\rho, \widetilde{\theta}_k)$ is the $(1-\rho)$-quantile of $H(X)$ with respect to $\widetilde{f}(\cdot, \widetilde{\theta}_k)$. By the definition of quantiles, we have

$$\begin{aligned} \widetilde{P}_{\widetilde{\theta}_k}\left(H(X) \geq \gamma_{k+1}(\rho, \widetilde{\theta}_k)\right) &\geq \rho, \quad \text{and} \\ \widetilde{P}_{\widetilde{\theta}_k}\left(H(X) \leq \gamma_{k+1}(\rho, \widetilde{\theta}_k)\right) &\geq 1 - \rho > 1 - \rho_k^*. \end{aligned} \tag{5.28}$$

It follows that $\widetilde{P}_{\widetilde{\theta}_k}\left(H(X) \leq \gamma_{k+1}(\rho, \widetilde{\theta}_k)\right) \leq \widetilde{P}_{\widetilde{\theta}_k}\left(H(X) < \bar{\gamma}_k + \frac{2\varepsilon}{3}\right) = 1 - \rho_k^*$ by the definition of ρ_k^*, which contradicts equation (5.28); thus we must have that if $\bar{\gamma}_k \leq H(x^*) -$

ε, then

$$\gamma_{k+1}(\rho, \widetilde{\theta}_k) \geq \bar{\gamma}_k + \frac{2\varepsilon}{3}, \quad \forall \rho \in (0, \rho_k^*).$$

Therefore by (5.27), $\exists \bar{\rho} \in \big(0, \min\{\rho_k, \lambda\mathcal{C}(\varepsilon, \theta_0)\}\big) \subseteq (0, \rho_k)$ such that $\gamma_{k+1}(\bar{\rho}, \widetilde{\theta}_k) \geq \bar{\gamma}_k + \frac{2\varepsilon}{3}$ whenever $\bar{\gamma}_k \leq H(x^*) - \varepsilon$. By Lemma 5.5.2, the distance from the sample $(1-\bar{\rho})$-quantile $\widetilde{\gamma}_{k+1}(\bar{\rho}, N_k)$ to the set of $(1-\bar{\rho})$-quantiles $\gamma_{k+1}(\bar{\rho}, \widetilde{\theta}_k)$ goes to zero as $k \to \infty$ w.p.1, thus $\exists \bar{\mathcal{K}} < \infty$ w.p.1 such that $\widetilde{\gamma}_{k'+1}(\bar{\rho}, N_{k'}) \geq \bar{\gamma}_k + \frac{\varepsilon}{2}\ \forall k' \geq \bar{\mathcal{K}}$.

Notice that from the MRAS$_1$ algorithm, if neither step $3a$ nor $3b$ is visited at the kth iteration, we will have $\rho_{k+1} = \rho_k$ and $\bar{\gamma}_{k+1} = \bar{\gamma}_k$. Thus, whenever $\bar{\gamma}_k \leq H(x^*) - \varepsilon$, w.p.1 step $3a/3b$ will be visited after a finite number of iterations. Furthermore, since the total number of visits to steps $3a$ and $3b$ is finite (i.e., bounded by $\frac{2[H(x^*)-\mathcal{M}]}{\varepsilon}$, where recall that $\mathcal{M}$ is a lower bound for $H(x)$), we conclude that there exists $\mathcal{K} < \infty$ w.p.1, such that

$$\bar{\gamma}_k > H(x^*) - \varepsilon, \quad \forall k \geq \mathcal{K} \quad w.p.1.$$

(2) From the MRAS$_1$ algorithm, it is easy to see that $\bar{\gamma}_{k+1} \geq \bar{\gamma}_k$, $\forall k = 0, 1, \ldots$. By part **(1)**, we have $\bar{\gamma}_{k+1} \geq H(x^*) - \varepsilon$, $\forall k \geq \mathcal{K}$ w.p.1. Thus, by the definition of $\widetilde{g}_{k+1}(x)$ (cf. (5.22)), it follows immediately that if $\Big\{x : H(x) \geq \bar{\gamma}_{k+1},\ x \in \Big\{X_1^k, \ldots, X_{N_k}^k\Big\}\Big\} \neq \emptyset$, then the support of $\widetilde{g}_{k+1}(x)$ satisfies $supp\,\{\widetilde{g}_{k+1}\} \subseteq \mathcal{O}_\varepsilon\ \forall k \geq \mathcal{K}$ w.p.1; otherwise if $\Big\{x : H(x) \geq \bar{\gamma}_{k+1},\ x \in \Big\{X_1^k, \ldots, X_{N_k}^k\Big\}\Big\} = \emptyset$, then $supp\,\{\widetilde{g}_{k+1}\} = \emptyset$. We now discuss these two cases separately.

Case 1. If $supp\,\{\widetilde{g}_{k+1}\} \subseteq \mathcal{O}_\varepsilon$, then we have $\{\Gamma(supp\,\{\widetilde{g}_{k+1}\})\} \subseteq \{\Gamma(\mathcal{O}_\varepsilon)\}$. Since $E_{\widetilde{g}_{k+1}}[\Gamma(X)]$ is the convex combination of $\Gamma(X_1^k), \ldots, \Gamma(X_{N_k}^k)$, it follows that

$$E_{\widetilde{g}_{k+1}}[\Gamma(X)] \in CONV\,\{\Gamma\,(supp\,\{\widetilde{g}_{k+1}\})\} \subseteq CONV\,\{\Gamma(\mathcal{O}_\varepsilon)\}\,.$$

Thus by A4$'$, A5, and Lemma 5.5.1,

$$E_{\widetilde{\theta}_{k+1}}\left[\Gamma(X)\right] \in CONV\left\{\Gamma(\mathcal{O}_\varepsilon)\right\}.$$

Case 2. If $supp\left\{\widetilde{g}_{k+1}\right\} = \emptyset$ (note that this could only happen if step 3c is visited), then from the algorithm, there exists some $\widehat{k} < k+1$ such that $\bar{\gamma}_{k+1} = \bar{\gamma}_{\widehat{k}}$ and $supp\left\{\widetilde{g}_{\widehat{k}}\right\} \neq \emptyset$. Without loss of generality, let $\widehat{k}$ be the largest iteration counter such that the preceding properties hold. Since $\bar{\gamma}_{\widehat{k}} = \bar{\gamma}_{k+1} > H(x^*) - \varepsilon\ \forall\, k \geq \mathcal{K}$ w.p.1, we have $supp\left\{\widetilde{g}_{\widehat{k}}\right\} \subseteq \mathcal{O}_\varepsilon$ w.p.1. By following the discussions in Case 1, it is clear that

$$E_{\widetilde{\theta}_{\widehat{k}}}\left[\Gamma(X)\right] \in CONV\left\{\Gamma(\mathcal{O}_\varepsilon)\right\},\ w.p.1.$$

Furthermore, since $\widetilde{\theta}_{\widehat{k}} = \widetilde{\theta}_{\widehat{k}+1} = \cdots = \widetilde{\theta}_{k+1}$ (see discussions in Chapter 5.5.1), we will again have

$$E_{\widetilde{\theta}_{k+1}}\left[\Gamma(X)\right] \in CONV\left\{\Gamma(\mathcal{O}_\varepsilon)\right\},\ \forall\, k \geq \mathcal{K}\ w.p.1.$$

(3) Define $\widehat{g}_{k+1}(x)$ as

$$\widehat{g}_{k+1}(x) := \frac{[S(H(x))]^k I_{\{H(x)\geq\bar{\gamma}_k\}}}{\int_{\mathcal{X}}[S(H(x))]^k I_{\{H(x)\geq\bar{\gamma}_k\}}\nu(dx)},\quad \forall\, k = 1, 2, \ldots,$$

where $\bar{\gamma}_k$ is defined as in MRAS$_1$. Note that since $\bar{\gamma}_k$ is a random variable, $\widehat{g}_{k+1}(x)$ is also a random variable. It follows that

$$E_{\widehat{g}_{k+1}}\left[\Gamma(X)\right] = \frac{\int_{\mathcal{X}}[\beta S(H(x))]^k I_{\{H(x)\geq\bar{\gamma}_k\}}\Gamma(x)\nu(dx)}{\int_{\mathcal{X}}[\beta S(H(x))]^k I_{\{H(x)\geq\bar{\gamma}_k\}}\nu(dx)}.$$

Let $\omega = (X_1^0, \ldots, X_{N_0}^0, X_1^1, \ldots, X_{N_1}^1, \ldots)$ be a particular sample path generated by the algorithm. For each ω, the sequence $\{\bar{\gamma}_k(\omega),\ k = 1, 2, \ldots\}$ is non-decreasing and each strict increase is lower bounded by $\varepsilon/2$. Thus, $\exists\, \widetilde{\mathcal{N}}(\omega) > 0$ such that $\bar{\gamma}_{k+1}(\omega) = \bar{\gamma}_k(\omega)\ \forall\, k \geq \widetilde{\mathcal{N}}(\omega)$. Now define $\Omega_1 := \{\omega : \lim_{k\to\infty} \bar{\gamma}_k(\omega) = H(x^*)\}$. By the definition of $\widetilde{g}_{k+1}(\cdot)$ (cf. (5.22)), for each $\omega \in \Omega_1$ we clearly have $\lim_{k\to\infty} E_{\widetilde{g}_k(\omega)}\left[\Gamma(X)\right] = \Gamma(x^*)$; thus, it follows from Lemma 5.5.1 that $\lim_{k\to\infty} E_{\widetilde{\theta}_k(\omega)}[\Gamma(X)] = \Gamma(x^*),\ \forall\,\omega \in \Omega_1$. The rest of the proof amounts to showing that the result also holds almost surely (a.s.) on the set Ω_1^c.

Since $\lim_{k\to\infty} \bar{\gamma}_k(\omega) = \bar{\gamma}_{\widetilde{\mathcal{N}}}(\omega) < H(x^*)\ \forall\,\omega \in \Omega_1^c$, we have by Fatou's lemma

$$\begin{aligned}\liminf_{k\to\infty} \int_{\mathcal{X}} [\beta S(H(x))]^k I_{\{H(x)\geq \bar{\gamma}_k\}} \nu(dx) &\geq \int_{\mathcal{X}} \liminf_{k\to\infty} [\beta S(H(x))]^k I_{\{H(x)\geq \bar{\gamma}_k\}} \nu(dx) \\ &> 0,\ \forall\,\omega \in \Omega_1^c, \end{aligned} \tag{5.29}$$

where the last inequality follows from the fact that $\beta S(H(x)) \geq 1\ \forall\, x \in \big\{x : H(x) \geq \max\{S^{-1}(\frac{1}{\beta}), \bar{\gamma}_{\widetilde{\mathcal{N}}}\}\big\}$ and assumption A1.

Since $f(x,\theta_0) > 0\ \forall\, x \in \mathcal{X}$, we have $\mathcal{X} \subseteq supp\{\widetilde{f}(\cdot,\widetilde{\theta}_k)\}\ \forall\, k$; thus

$$E_{\widehat{g}_{k+1}}[\Gamma(X)] = \frac{\widetilde{E}_{\widetilde{\theta}_k}\big[\beta^k \widetilde{S}_k(H(X)) I_{\{H(X)\geq\bar{\gamma}_k\}}\Gamma(X)\big]}{\widetilde{E}_{\widetilde{\theta}_k}\big[\beta^k \widetilde{S}_k(H(X)) I_{\{H(X)\geq\bar{\gamma}_k\}}\big]},\ \forall\, k = 1,2,\ldots,$$

where $\widetilde{S}_k(H(x)) := [S(H(x))]^k / \widetilde{f}(x,\widetilde{\theta}_k)$. We now show that $E_{\widetilde{g}_{k+1}}[\Gamma(X)] \to E_{\widehat{g}_{k+1}}[\Gamma(X)]$ a.s. on Ω_1^c as $k\to\infty$. Since we are only interested in the limiting behavior of $E_{\widetilde{g}_{k+1}}[\Gamma(X)]$, it is sufficient to show that

$$\frac{\frac{1}{N_k}\sum_{i=1}^{N_k} \beta^k \widetilde{S}_k(H(X_i^k)) I_{\{H(X_i^k)\geq\bar{\gamma}_{k+1}\}}\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k} \beta^k \widetilde{S}_k(H(X_i^k)) I_{\{H(X_i^k)\geq\bar{\gamma}_{k+1}\}}} \longrightarrow E_{\widehat{g}_{k+1}}[\Gamma(X)]\ \text{ a.s. on } \Omega_1^c,$$

where and hereafter, whenever $\big\{x : H(x) \geq \bar{\gamma}_{k+1},\ x \in \{X_1^k,\ldots,X_{N_k}^k\}\big\} = \emptyset$, we define $\frac{0}{0} = 0$.

For brevity, we use the following shorthand notations:

$$\widehat{Y}^k := \widetilde{E}_{\widetilde{\theta}_k}[\beta^k \widetilde{S}_k(H(X)) I_{\{H(X)\geq\bar{\gamma}_k\}}],\quad \widehat{Y}_\Gamma^k := \widetilde{E}_{\widetilde{\theta}_k}[\beta^k \widetilde{S}_k(H(X)) I_{\{H(X)\geq\bar{\gamma}_k\}}\Gamma(X)],$$

$$\bar{Y}_i^k := \beta^k \widetilde{S}_k(H(X_i^k)) I_{\{H(X_i^k)\geq\bar{\gamma}_{k+1}\}},\quad \widehat{Y}_i^k := \beta^k \widetilde{S}_k(H(X_i^k)) I_{\{H(X_i^k)\geq\bar{\gamma}_k\}}.$$

We also let $\mathcal{T}_\varepsilon := \lceil \frac{2[H(x^*)-\mathcal{M}]}{\varepsilon} \rceil$. Note that the total number of visits to step $3a$ and $3b$ of MRAS$_1$ is bounded by $\mathcal{T}_\varepsilon$, thus for any $k > \mathcal{T}_\varepsilon$, the total number of visits to step $3c$ is greater than $k - \mathcal{T}_\varepsilon$.

We have

$$\frac{\frac{1}{N_k}\sum_{i=1}^{N_k}\beta^k\widetilde{S}_k(H(X_i^k))I_{\{H(X_i^k)\geq\bar{\gamma}_{k+1}\}}\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k}\beta^k\widetilde{S}_k(H(X_i^k))I_{\{H(X_i^k)\geq\bar{\gamma}_{k+1}\}}} \quad - \quad E_{\widehat{g}_{k+1}}[\Gamma(X)] =$$
$$\left(\frac{\frac{1}{N_k}\sum_{i=1}^{N_k}\bar{Y}_i^k\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k}\bar{Y}_i^k} - \frac{\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k}\right) \quad + \quad \left(\frac{\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k} - \frac{\widehat{Y}_\Gamma^k}{\widehat{Y}^k}\right).$$

Since for each $\omega\in\Omega_1^c$, $\bar{\gamma}_{k+1}(\omega)=\bar{\gamma}_k(\omega)\;\forall\,k\geq\widetilde{\mathcal{N}}(\omega)$, it is straightforward to see that the first term

$$\frac{\frac{1}{N_k}\sum_{i=1}^{N_k}\bar{Y}_i^k\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k}\bar{Y}_i^k} - \frac{\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k} = 0, \quad \forall\,k\geq\widetilde{\mathcal{N}}(\omega),\;\forall\,\omega\in\Omega_1^c. \tag{5.30}$$

To show that the second term also converges to zero, we denote by $\mathcal{V}_k$ the event $\mathcal{V}_k=\{\bar{\gamma}_k>H(x^*)-\varepsilon\}$. For any $\zeta>0$, we also let $\mathcal{C}_k$ be the event $\mathcal{C}_k=\{\big|\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k-\widehat{Y}^k\big|>\zeta\}$. We have

$$\begin{aligned} P(\mathcal{C}_k\ i.o.) &= P(\{\mathcal{C}_k\cap\mathcal{V}_k\}\cup\{\mathcal{C}_k\cap\mathcal{V}_k^c\}\ i.o.) \\ &= P(\mathcal{C}_k\cap\mathcal{V}_k\ i.o.),\ \text{since } P(\mathcal{V}_k^c\ i.o.)=0 \text{ by part } \mathbf{(1)}. \end{aligned} \tag{5.31}$$

It is easy to see that conditional on $\widetilde{\theta}_k$ and $\bar{\gamma}_k$, $\widehat{Y}_1^k,\ldots,\widehat{Y}_{N_k}^k$ are i.i.d. and $E[\widehat{Y}_i^k|\widetilde{\theta}_k,\bar{\gamma}_k]=\widehat{Y}^k\;\forall\,i$. Furthermore, by assumption A3$'$, conditional on the event $\mathcal{V}_k$, the support $[a_k,b_k]$ of the random variable $\widehat{Y}_i^k$ satisfies $[a_k,b_k]\subseteq\left[0,\frac{(\beta S^*)^k}{\lambda f_*}\right]$. Therefore, we have from the Hoeffding inequality ([40]),

$$\begin{aligned} P\big(\mathcal{C}_k\,|\mathcal{V}_k,\widetilde{\theta}_k=\theta,\bar{\gamma}_k=\gamma\big) &= P\Big(\big|\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k-\widehat{Y}^k\big|>\zeta\,\Big|\mathcal{V}_k,\widetilde{\theta}_k=\theta,\bar{\gamma}_k=\gamma\Big), \\ &\leq 2\exp\Big(\frac{-2N_k\zeta^2}{(b_k-a_k)^2}\Big), \\ &\leq 2\exp\Big(\frac{-2N_k\zeta^2[\lambda f_*]^2}{(\beta S^*)^{2k}}\Big)\;\;\forall\,k=1,2\ldots. \end{aligned} \tag{5.32}$$

Since

$$\begin{aligned} P(\mathcal{C}_k \cap \mathcal{V}_k) &= \int_{\theta,\gamma} P(\mathcal{C}_k \cap \mathcal{V}_k \,|\widetilde{\theta}_k = \theta, \bar{\gamma}_k = \gamma) f_{\widetilde{\theta}_k,\bar{\gamma}_k}(d\theta, d\gamma), \\ &= \int_{\theta,\mathcal{V}_k} P(\mathcal{C}_k \,|\mathcal{V}_k, \widetilde{\theta}_k = \theta, \bar{\gamma}_k = \gamma) f_{\widetilde{\theta},\bar{\gamma}_k}(d\theta, d\gamma), \end{aligned}$$

where $f_{\widetilde{\theta}_k,\bar{\gamma}_k}(\cdot,\cdot)$ is the joint distribution of random variables $\widetilde{\theta}_k$ and $\bar{\gamma}_k$, we have by (5.32),

$$\begin{aligned} P(\mathcal{C}_k \cap \mathcal{V}_k) &\leq 2\exp\left(\frac{-2N_k\zeta^2[\lambda f_*]^2}{(\beta S^*)^{2k}}\right), \\ &\leq 2\exp\left(\frac{-2(\alpha^{k-\mathcal{T}_\varepsilon}N_0)\zeta^2[\lambda f_*]^2}{(\beta S^*)^{2k}}\right) \;\; \forall\, k \geq \mathcal{T}_\varepsilon, \\ &= 2\exp\left(\frac{-2N_0\zeta^2\lambda^2 f_*^2}{\alpha^{\mathcal{T}_\varepsilon}}\Big(\frac{\alpha}{(\beta S^*)^2}\Big)^k\right) \;\; \forall\, k \geq \mathcal{T}_\varepsilon, \end{aligned}$$

Since $\alpha/(\beta S^*)^2 > 1$ (by assumption), it follows that

$$\lim_{k\to\infty} P(\mathcal{C}_k \cap \mathcal{V}_k) = 0.$$

Furthermore, since $e^{-x} < 1/x \;\forall\; x > 0$ we have

$$P(\mathcal{C}_k \cap \mathcal{V}_k) < \frac{\alpha^{\mathcal{T}_\varepsilon}}{N_0\zeta^2\lambda^2 f_*^2}\Big(\frac{(\beta S^*)^2}{\alpha}\Big)^k \;\; \forall\, k \geq \mathcal{T}_\varepsilon,$$

and because $(\beta S^*)^2/\alpha < 1$, we have

$$\sum_{k=0}^{\infty} P(\mathcal{C}_k \cap \mathcal{V}_k) < \mathcal{T}_\varepsilon + \frac{\alpha^{\mathcal{T}_\varepsilon}}{N_0\zeta^2\lambda^2 f_*^2}\sum_{k=\mathcal{T}_\varepsilon}^{\infty}\Big(\frac{(\beta S^*)^2}{\alpha}\Big)^k < \infty.$$

Finally by the Borel-Cantelli lemma and (5.31),

$$P(\mathcal{C}_k \;i.o) = P(\mathcal{C}_k \cap \mathcal{V}_k \;i.o.) = 0.$$

Since this holds for any $\zeta > 0$, we have $\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k \to \widehat{Y}^k$ w.p.1.

By following the same argument as before, we can also show that $\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k\Gamma(X_i^k) \rightarrow \widehat{Y}_\Gamma^k$ w.p.1. And since $\lim_{k\rightarrow\infty}\widehat{Y}^k > 0\ \forall\,\omega \in \Omega_1^c$ (i.e., (5.29)), we have

$$\frac{\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k}\widehat{Y}_i^k} \rightarrow \frac{\widehat{Y}_\Gamma^k}{\widehat{Y}^k} \text{ as } k \rightarrow \infty \text{ a.s. on } \Omega_1^c.$$

By the definition of $\widetilde{g}_{k+1}(\cdot)$, the above result together with (5.30) suggests that

$$E_{\widetilde{g}_k}\left[\Gamma(X)\right] \rightarrow E_{\widehat{g}_k}\left[\Gamma(X)\right] \quad \text{as } k \rightarrow \infty \text{ a.s. on } \Omega_1^c.$$

Thus, in conclusion, we have

$$E_{\widetilde{g}_k}\left[\Gamma(X)\right] \rightarrow E_{\widehat{g}_k}\left[\Gamma(X)\right] \text{ as } k \rightarrow \infty \text{ w.p.1.}$$

On the other hand, by A1, A2, and following the proof of Theorem 5.3.1, it is not difficult to show that

$$E_{\widehat{g}_k}\left[\Gamma(X)\right] \rightarrow \Gamma(x^*) \quad \text{as } k \rightarrow \infty \text{ w.p.1.}$$

Hence by Lemma 5.5.1, we have

$$\lim_{k\rightarrow\infty} E_{\widetilde{\theta}_k}\left[\Gamma(X)\right] = \lim_{k\rightarrow\infty} E_{\widetilde{g}_k}\left[\Gamma(X)\right] = \Gamma(x^*) \quad \text{w.p.1.}$$

∎

The following results are now immediate.

Corollary 5.5.2 (Multivariate Normal) *For continuous optimization problems in $\Re^n$, if multivariate normal p.d.f.'s are used in MRAS$_1$, i.e.,*

$$f(x,\widetilde{\theta}_k) = \frac{1}{\sqrt{(2\pi)^n|\widetilde{\Sigma}_k|}}\exp\Big(-\frac{1}{2}(x-\widetilde{\mu}_k)^T\widetilde{\Sigma}_k^{-1}(x-\widetilde{\mu}_k)\Big),$$

$\varepsilon > 0$, $\alpha > (\beta S^*)^2$, and assumptions A1, A2, A3′, and A4′ are satisfied, then

$$\lim_{k\rightarrow\infty}\widetilde{\mu}_k = x^*, \quad \text{and} \quad \lim_{k\rightarrow\infty}\widetilde{\Sigma}_k = 0_{n\times n} \quad \text{w.p.1.}$$

Corollary 5.5.3 (Independent Univariate) *If the components of the random vector $X = (X_1, X_2, \ldots, X_n)$ are independent, each with a univariate p.d.f./p.m.f. of the form*

$$f(x_i, \vartheta_i) = \exp(x_i\vartheta_i - K(\vartheta_i))h(x_i), \ \vartheta_i \in \Re, \forall\, i = 1, \ldots, n,$$

$\varepsilon > 0$, $\alpha > (\beta S^*)^2$, *and assumptions* $A1$, $A2$, $A3'$, $A4'$, *and* $A5$ *are satisfied, then*

$$\lim_{k\to\infty} E_{\widetilde{\theta}_k}[X] = x^* \quad w.p.1, \quad \textit{where}\ \ \widetilde{\theta}_k := (\vartheta_1^k, \ldots, \vartheta_n^k).$$

5.6 Numerical Examples

In this Chapter, we illustrate the performance of the MRAS method for both continuous and combinatorial optimization problems. In the former case, we test the algorithm on various functions that are well-known in global optimization and compare its performance with that of the standard CE method. In the latter case, we apply the algorithm to several Asymmetric Traveling Salesman Problems (ATSP), which are typical representatives of NP-hard combinatorial optimization problems.

Remark 5.6.1 *It is not our primary intention here to compare our algorithm with the CE method and EDAs. A comprehensive comparison of different methods is beyond the scope of this research. Our main goal here is to propose a novel algorithm with provable convergence, and show that the algorithm is promising in solving some difficult optimization problems. The performance of the CE method on continuous functions can be found in, e.g., [51], [66]. Its performance on various ATSP instances can be found in, e.g., [26], [67].*

We now discuss some implementation issues of the MRAS$_1$ algorithm.

1. Since all examples considered in this Chapter are minimization problems, whereas

MRAS was presented in a maximization context, the following modifications are required:

- $S(\cdot)$ needs to be initialized as a strictly decreasing function instead of strictly increasing. Throughout this Chapter, we take

$$S(H(x)) := \exp\{-rH(x)\}, \text{ where } r \text{ is a positive constant.}$$

- The sample $(1-\rho)$-quantile $\widetilde{\gamma}_{k+1}$ will now be calculated by first ordering the sample performances $H(X_i^k),\ i = 1, \ldots, N_k$ from largest to smallest, and then taking the $\lceil(1-\rho)N_k\rceil$th order statistic.
- We need to replace the "$\geq$" operator with "$\leq$" operator in equation (5.21).
- The inequalities at step 3 need to be replaced with

$$\widetilde{\gamma}_{k+1}(\rho_k, N_k) \leq \bar{\gamma}_k - \frac{\varepsilon}{2}, \text{ and } \widetilde{\gamma}_{k+1}(\bar{\rho}, N_k) \leq \bar{\gamma}_k - \frac{\varepsilon}{2},$$

respectively.

2. Similar to CE, a smoothed parameter updating procedure (cf. e.g., [26], [66]) is used in actual implementation, i.e., first a smoothed parameter vector $\widehat{\theta}_{k+1}$ is computed at each iteration k according to

$$\widehat{\theta}_{k+1} := \upsilon\,\widetilde{\theta}_{k+1} + (1-\upsilon)\widehat{\theta}_k, \quad \forall\ k = 0, 1, \ldots, \text{ and } \widehat{\theta}_0 := \widetilde{\theta}_0,$$

where $\widetilde{\theta}_{k+1}$ is the parameter vector computed at step 4 of MRAS$_1$, and $\upsilon \in (0,1]$ is the smoothing parameter; then $f(x, \widehat{\theta}_{k+1})$ (instead of $f(x, \widetilde{\theta}_{k+1})$) is used in step 1 to generate new samples. It is important to note that this modification will not affect the theoretical convergence of our approach.

3. In practice, different stopping criteria can be used. The simplest method is to stop the algorithm when a predefined maximum number of iterations is reached, or when the total computational budget is exhausted. In the numerical experiments, a mixed stopping rule is used: We stop the algorithm either when no significant improvement in $\bar{\gamma}_k$ is obtained for several consecutive iterations or when the sample size at a single iteration exceeds some predefined threshold, i.e., as soon as either one of the following two conditions is satisfied at iteration k:

   (1) $\max_{1 \leq i \leq d} |\bar{\gamma}_k - \bar{\gamma}_{k+i}| \leq \tau$;

   (2) $N_k > N_{\max}$;

   where $\tau > 0$ is a predefined tolerance level, d is a positive integer, and $N_{\max}$ is the maximum number of samples allowed per iteration.

4. Another practical issue is that in order to obtain a valid estimate $\widetilde{\theta}_{k+1}$ at each iteration of MRAS$_1$, we must make sure that enough samples are used in parameter updating. This can be achieved by using an additional parameter N_{min}, and performing the update (5.21) only when the number of the elite samples (i.e., those samples having performances better than the threshold $\bar{\gamma}_{k+1}$) is greater than N_{min}. In effect, this is equivalent to searching $\bar{\rho}$ from (ρ_{min}, ρ_k) instead of $(0, \rho_k)$ at step 3 of MRAS$_1$, where $\rho_{min} := N_{min}/N_k \to 0$ as $k \to \infty$.

5.6.1 Continuous Optimization

In our preliminary experiments, we take the family of parameterized p.d.f.'s to be multivariate normal p.d.f.'s. Initially, a mean vector μ_0 and a covariance matrix Σ_0 are specified; then at each iteration k of the algorithm, new parameters $\widetilde{\mu}_{k+1}$ and $\widetilde{\Sigma}_{k+1}$ are updated according to the respective stochastic counterparts of equations (5.16) and (5.17). By Corollary 5.5.2, the sequence of mean vectors $\{\widetilde{\mu}_k\}$ will converge to the optimal solution x^*, and the sequence of covariance matrices $\{\widetilde{\Sigma}_k\}$ to the zero matrix. Throughout this Chapter, we will use $\widetilde{\mu}_k$ to represent the current best solution found at iteration k.

The following five functions $\{H_i,\ i = 1, \ldots, 5\}$ are used to test the algorithm.

(1) Quadratic function

$$H_1(x) = \sum_{i=1}^{3} x_i^2, \quad \text{where } x = (x_1, x_2, x_3).$$

The function has a unique global minimum $f(0, 0, 0) = 0$.

(2) Two-dimensional Rosenbrock function

$$H_2(x) = 100(x_1^2 - x_2)^2 + (1 - x_1^2), \quad \text{where } x = (x_1, x_2).$$

The function has the reputation of being difficult to minimize and is widely used to test the performance of different optimization algorithms. It has a global minimum $f(1, 1) = 0$.

(3) Shekel's Foxholes

$$H_3(x) = \frac{1}{0.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{j,i})^6}},$$

where $a_{j,1} = \{-32, -16, 0, 16, 32, -32, -16, 0, 16, 32, -32, -16, 0, 16, 32, -32, -16, 0, 16, 32, -32, -16, 0, 16, 32\}$,

$a_{j,2} = \{-32, -32, -32, -32, -32, -16, -16, -16, -16, -16, 0, 0, 0, 0, 0, 16, 16, 16, 16, 16, 32, 32, 32, 32, 32\}$, and $x = (x_1, x_2)$. The function has 24 local minima and one global minimum $f(-32, -32) \approx 0.998004$.

(4) Corana's Parabola

$$H_4(x) = \sum_{i=1}^{4} \begin{cases} 0.15[0.05\,\mathrm{sgn}(z_i) - z_i]^2 h_j & \text{if } |x_i - z_i| < 0.05, \\ h_i x_i^2 & \text{otherwise,} \end{cases}$$

where

$$z_i = 0.2 \left\lfloor \left| \frac{x_i}{2} \right| + 0.49999 \right\rfloor \mathrm{sgn}(x_i),$$

$h = \{1, 1000, 10, 100\}$, and $x = (x_1, x_2, x_3, x_4)$. In the region $-1000 < x_i < 1000$, $i = 1, 2, 3, 4$, the above function has more than 10^{20} local minima, which is very difficult to minimize. It has a global minimum $f(0, 0, 0, 0) = 0$.

(5) Goldstein-Price function

$$\begin{aligned} H_5(x) = \ & (1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)) \\ & (30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)), \end{aligned}$$

where $x = (x_1, x_2)^T$. The function has four local minima and a global minimum $f(0, -1) = 3$.

For all five problems, the same set of parameters is used to test MRAS: $\varepsilon = 10^{-5}$, initial sample size $N_0 = 100$, $\rho_0 = 0.2$, $\lambda = 0.02$, $\alpha = 1.5$, $r = 0.1$, the stopping control parameters $d = 5$, $\tau = 10^{-5}$, $N_{max} = 50000$, $N_{min} = 5n$, and the smoothing parameter $\upsilon = 0.5$. The initial mean vector μ_0 is a n-by-1 vector of all $10s$, and Σ_0 is a n-by-n diagonal matrix with all diagonal elements equal to 200, where recall that n is the dimension of the problem.

Table 5.1 shows the performance of the algorithm on the five test functions. For each function, we performed 50 independent replication runs of the algorithm, and the means and standard errors are reported in the table, where N_{total} is the total number of function evaluations, ρ_{final} is the final value of ρ, and $\bar{H}_i^*$ is the averaged value of the

H_i	N_{total} (std)	ρ_{final} (std)	$\bar{H}_i^*$ (std)	$H_i(x^*)$	M_ε
H_1	4.38e+03(6.77e+01)	0.13(6.21e-03)	9.86e-09(1.12e-09)	0	50
H_2	1.21e+04(4.89e+02)	0.04(2.40e-03)	2.29e-09(3.13e-10)	0	50
H_3	2.17e+04(7.16e+02)	0.02(1.11e-03)	2.40(4.15e-01)	0.998	37
H_4	7.43e+03(1.61e+02)	0.14(4.19e-03)	0.00(0.00e-00)	0	50
H_5	5.81e+03(1.40e+02)	0.11(5.88e-03)	3.00(5.30e-10)	3	50

Table 5.1: Performance of MRAS on five test functions, based on 50 independent replication runs. The standard errors are in parentheses.

function $H_i(\cdot)$ at the best solution visited by the algorithm. The optimal value $H_i(x^*)$ is included for reference, and M_ε indicates the number of runs out of 50 trials in which an ε-optimal solution was found. The algorithm performs quite well in most cases, except for H_3, where only 37 ε-optimal solutions were found. H_3 represents a class of continuous optimization problems that are extremely difficult to solve for most *model-based* sampling approaches. A graphical representation of the function H_3 is given in Figure 5.3. Notice that the function values at the 25 "holes" (local minima) are very close to each other; thus in order to locate the global optimal solution, the algorithm must make sure that samples are drawn from the right "hole", and there must be enough samples to fall in this "hole" to guarantee that the parameter vectors are updated in the right direction.

For comparison purposes, we also applied the CE method to the above five test functions, where we have used the multivariate normal p.d.f. with independent components (cf. e.g., [51] for detailed algorithm description and implementation issues). We have tested different sets of parameters (i.e., different (N,ρ) combinations); the results reported in Table 5.2 are based on the following "good" parameter settings: sample size

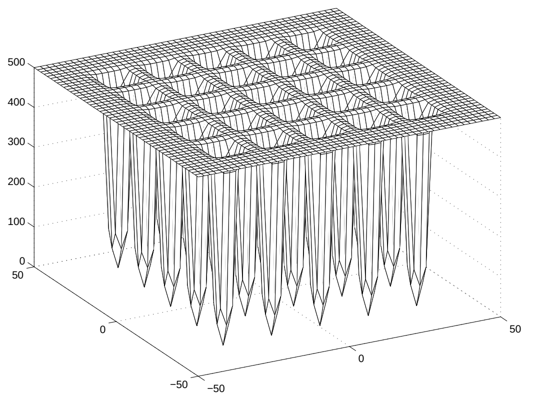


Figure 5.3: Shekel's Foxholes, where $-50 \leq x_i \leq 50,\ i = 1, 2$.

$N = 1000$ (recall that the CE method is non-adaptive, so the same number of samples will be generated at each iteration), $\rho = 0.005$, smoothing parameter $\upsilon = 0.7$, and the algorithm is stopped either when there exists $k > 0$ such that $\max_{1 \leq i \leq 5} |\widehat{\gamma}_k - \widehat{\gamma}_{k+i}| \leq 10^{-5}$ or when the total number of samples generated exceeds 2×10^5, where $\widehat{\gamma}_k$ is the sample $(1-\rho)$-quantile generated at the kth iteration of CE.

Again, the mean vector μ_0 is initialized as a n-by-1 vector of all 10s and the variances are taken to be a n-by-1 vector with all elements equal to 200.

H_i	N_{total} (std)	$\bar{H}_i^*$ (std)	$H_i(x^*)$	M_ε
H_1	1.69e+04(1.73e+02)	4.94e-05(5.13e-06)	0	7
H_2	1.72e+04(1.18e+02)	1.92e-05(2.93e-06)	0	24
H_3	1.05e+04(1.08e+02)	8.83(2.54e-01)	0.998	0
H_4	5.84e+04(6.06e+03)	1.35e-03(4.21e-04)	0	38
H_5	1.89e+05(4.77e+03)	3.00(5.63e-05)	3	0

Table 5.2: Performance of the standard CE method on five test functions, based on 50 independent runs. The standard errors are in parentheses.

From Tables 5.1 and 5.2, we see that MRAS uses fewer samples than CE does, but produces more accurate solutions. In general, the sequence $\{\widehat{\gamma}_k\}$ generated by CE may often converge quickly to a small neighborhood of $H(x^*)$; however, since no sample performances are used in parameter updating, (i.e., the top $\rho\%$ samples are all considered to be of the same importance regardless of their sample performances), the future search will be biased toward the region that has been sampled most. In particular, for the H_3 case, since the function values at different local minima are very close to each other, even if the "hole" with the global minimum has been sampled during the search process, CE still cannot distinguish the global minimum from the other local minima; instead CE will easily get stuck in the "hole" that has been sampled the most. As a result, we see that the algorithm gets trapped in local minima in all 50 trials. In contrast, the parameter updating procedure in MRAS is weighted by the performance function so that better samples will have more positive influence on the updating process. Consequently, the searches in MRAS will be biased toward the region containing more promising samples.

Table 5.3 gives the performance of CE and MRAS on function H_3 using different sample sizes and ρ values (all other parameters are the same as before). Test results indicate that increasing the samples size in CE has little effect on the quality of the resultant solutions. We see that the algorithm consistently gets stuck in local minima in repeated experiments. On the other hand, for MRAS with $N_0 = 200$, ε-optimal solutions were found in more than 90% of the total simulation runs; whereas for the $N_0 \geq 500$ cases, ε-optimal solutions were found in all 50 runs.

To illustrate the performance of the algorithm on high-dimensional problems, we also applied MRAS$_1$ to the following benchmark problems, which have been previously studied in e.g., [24], [61], [88], and [51]. Functions H_6 is a 4-dimensional problem which has

method	parameters	N_{total} (std)	$\bar{H}_3^*$ (std)	M_ε
CE	N=1000, ρ=0.1	1.47e+04(1.39e+02)	18.29(0.18)	0
	N=1000, ρ=0.01	1.13e+04(1.08e+02)	11.90(0.27)	0
	N=2000, ρ=0.1	2.91e+04(2.07e+02)	18.30(0.09)	0
	N=2000, ρ=0.01	2.25e+04(1.70e+02)	12.27(0.19)	0
	N=2000, ρ=0.005	2.14e+04(2.25e+02)	8.43(0.21)	0
	N=5000, ρ=0.1	7.19e+04(3.47e+02)	18.30(8.07e-11)	0
	N=5000, ρ=0.01	5.70e+04(3.78e+02)	12.52(0.14)	0
	N=5000, ρ=0.001	4.87e+04(6.67e+02)	5.61(0.32)	0
	N=10000, ρ=0.1	1.42e+05(6.10e+02)	18.30(4.80e-11)	0
	N=10000, ρ=0.01	1.12e+05(6.19e+02)	12.67(1.96e-12)	0
	N=10000, ρ=0.001	1.01e+05(1.39e+03)	4.80(0.32)	0
MRAS$_1$	N_0=200, ρ_0=0.2	2.27e+04(6.77e+02)	1.14(0.06)	45
	N_0=200, ρ_0=0.1	2.17e+04(7.14e+02)	1.08(0.05)	47
	N_0=500, ρ_0=0.2	3.01e+04(6.67e+02)	0.998(3.41e-11)	50
	N_0=500, ρ_0=0.1	2.76e+04(8.70e+02)	0.998(3.92e-11)	50
	N_0=1000, ρ_0=0.2	5.62e+04(8.30e+02)	0.998(3.41e-11)	50
	N_0=1000, ρ_0=0.1	4.31e+04(8.46e+02)	0.998(3.81e-11)	50

Table 5.3: Performance of CE and MRAS on test function H_3, based on 50 independent simulation runs. The standard errors are in parentheses. The optimum $H_3(x^*) \approx 0.998004$.

only a few local optima; however, the minima are separated by plateaus and are relatively far apart. Functions H_7 and H_8 are 20-dimensional badly-scaled problems. Functions H_9

and H_{10} are highly multimodal and the number of local optima increases exponentially with the problem dimension. Function H_{11} is both badly scaled and highly multimodal. The graphical representations of some of these functions in two dimensions are plotted in Figure 5.4.

(6) Shekel's function

$$H_6(x) = \sum_{i=1}^{5} \Big((x - a_i)^T (x - a_i) + c_i \Big)^{-1},$$

where $x = (x_1, x_2, x_3, x_4)^T$, $a_1 = (4,4,4,4)^T$, $a_2 = (1,1,1,1)^T$, $a_3 = (8,8,8,8)^T$, $a_4 = (6,6,6,6)^T$, $a_5 = (3,7,3,7)^T$, and $c = (0.1, 0.2, 0.2, 0.4, 0.4)$. The global minimizer $x^* \approx (4,4,4,4)^T$, and $H_6(x^*) \approx -10.153$.

(7) Rosenbrock function

$$H_7(x) = \sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2,$$

where $n = 20$. The global minimum is $x^* = (1, \dots, 1)^T$, and $H_7(x^*) = 0$.

(8) Powel singular function

$$H_8(x) = \sum_{i=2}^{n-2} \big[(x_{i-1} + 10x_i)^2 + 5(x_{i+1} - x_{i+2})^2 + (x_i - 2x_{i+1})^4 + 10(x_{i-1} - x_{i+2})^4 \big],$$

where $n = 20$, $x^* = (0, \dots, 0)^T$, and $H_8(x^*) = 0$.

(9) Trigonometric function

$$H_9(x) = 1 + \sum_{i=1}^{n} 8 \sin^2 \big(7(x_i - 0.9)^2 \big) + 6 \sin^2 \big(14(x_i - 0.9)^2 \big) + (x_i - 0.9)^2,$$

where $n = 20$, $x^* = (0.9, \dots, 0.9)^T$, and $H_9(x^*) = 1$.

(10) Griewank function

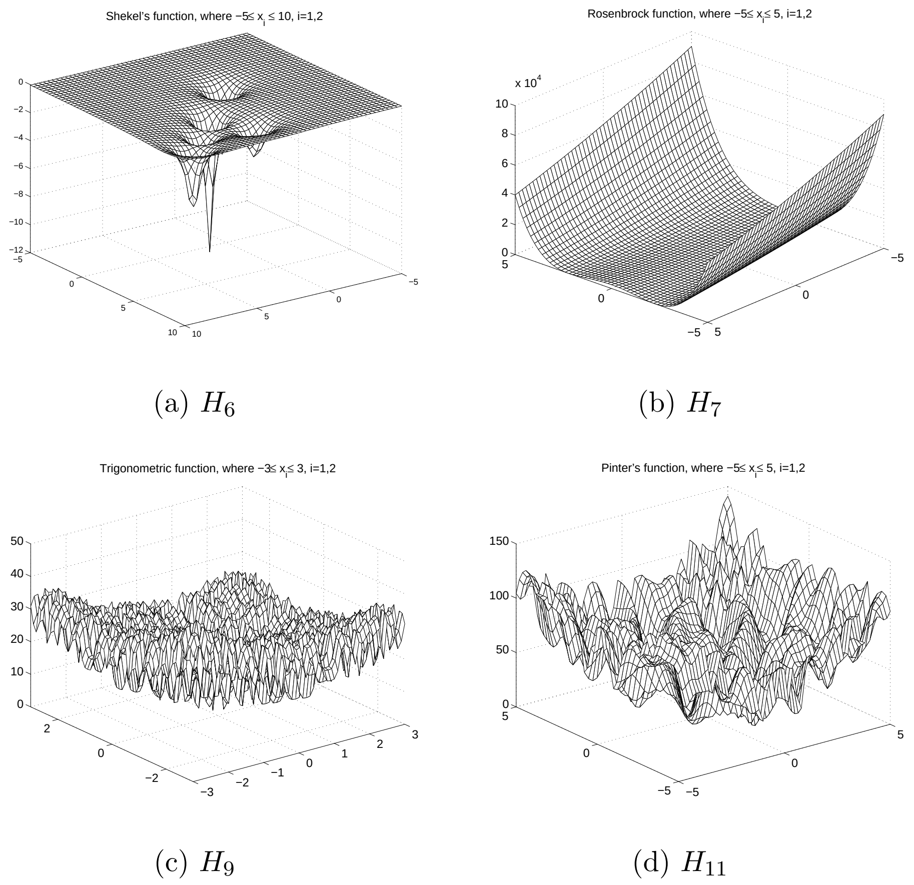
$$H_{10}(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1,$$

where $n = 20$, $x^* = (0, \dots, 0)^T$, and $H_{10}(x^*) = 0$.

(11) Pintér's function

$$
\begin{aligned}
H_{11}(x) &= \sum_{i=1}^{n} i x_i^2 + \sum_{i=1}^{n} 20 i \sin^2\big(x_{i-1}\sin x_i - x_i + \sin x_{i+1}\big) \\
&\quad + \sum_{i=1}^{n} i \log_{10}\big(1 + i(x_{i-1}^2 - 2x_i + 3x_{i+1} - \cos x_i + 1)^2\big),
\end{aligned}
$$

where $x_0 = x_n$, $x_{n+1} = x_1$, $n = 20$, $x^* = (0, \ldots, 0)^T$, and $H_{11}(x^*) = 0$.



(a) H_6 (b) H_7

(c) H_9 (d) H_{11}

Figure 5.4: Selected test problems in two dimensions, (a) H_6: Shekel; (b) H_7: Rosenbrock; (c) H_9: Trigonometric; (d) H_{11}: Pintér.

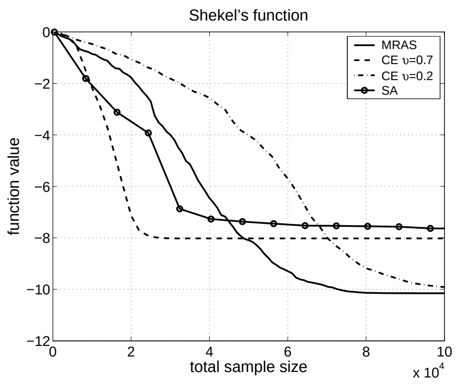
For all problems H_6–H_{11}, the same set of parameters is used to test MRAS$_1$: $\varepsilon = 10^{-5}$, initial sample size $N_0 = 1000$, $\rho_0 = 0.1$, $\lambda = 0.01$, $\alpha = 1.1$, $r = 10^{-4}$, smoothing parameter $\upsilon = 0.2$, and $N_{min} = 5n$. The initial mean vector μ_0 is a n-by-1 vector with each component randomly selected from the interval $[-50, 50]$ according to the uniform

Test	MRAS$_1$		CE ($\upsilon = 0.7$)		CE ($\upsilon = 0.2$)		SA	
Prob.	$\bar{H}_i^*$ (stderr)	M_ε	$\bar{H}_i^*$ (stderr)	M_ε	$\bar{H}_i^*$ (stderr)	M_ε	$\bar{H}_i^*$ (stderr)	M_ε
H_6	-10.15(3e-7)	50	-8.0(0.5)	34	-9.9(0.13)	0	-7.3(0.4)	2
H_7	11.8(0.5)	0	27.9(3.43)	0	15.9(2e-02)	0	203.7(11.3)	0
H_8	3e-10(2e-11)	50	1e+4(4e+3)	3	3e-6(2e-7)	50	65.9(3.0)	0
H_9	1.6(0.13)	24	1.0(00e-00)	50	1.0(6e-12)	50	65.2(1.22)	0
H_{10}	4e-3(7e-4)	28	2e-4(2e-4)	49	2e-12(4e-13)	50	0.15(0.04)	0
H_{11}	3e-9(6e-10)	50	2.3(1e-3)	0	6e-4(3e-05)	0	1.7e+3(51)	0

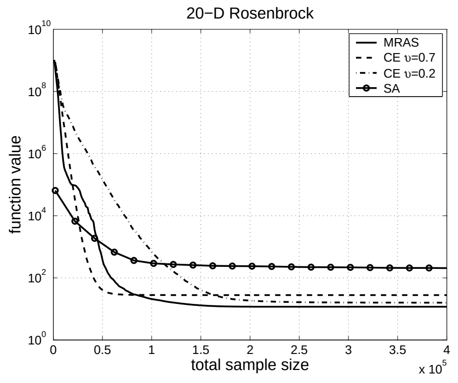
Table 5.4: Performance of different algorithms on benchmark problems $H_6 - H_{11}$, based on 50 independent runs. The standard errors are in parentheses.

distribution, and Σ_0 is a n-by-n diagonal matrix with all diagonal elements equal to 500.

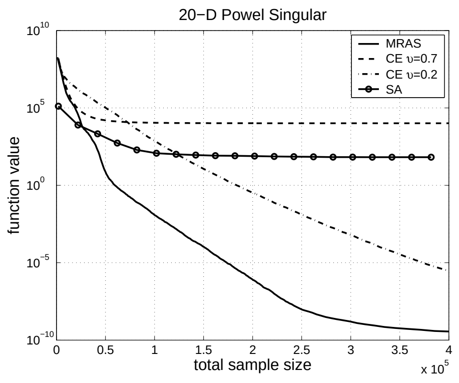
For comparison purposes, we also applied the CE method and the SA algorithm to the above test functions. For CE, we have used the univariate normal p.d.f. with parameter values suggested in [51]: sample size $N = 2000$, $\rho = 0.01$, smoothing parameter $\upsilon = 0.7$. Again, the initial mean vector μ_0 is randomly selected from $[-50, 50]^n$ according to the uniform distribution, and Σ_0 is a n-by-n diagonal matrix with all elements equal to 500. We found empirically that the above parameters work well for some functions, but in some other cases, the variance matrices in CE may converge too quickly to the zero matrix, which freezes the algorithm at some low quality solutions. To address this issue, for each problem, we also tried CE with different values of the smoothing parameter. In the numerical results reported below, we have used a smaller smoothing parameter value $\upsilon = 0.2$, which gives reasonable performance for all test cases. For SA, we have used the parameters suggested in [24]: initial temperature $T = 50000$, temperature reduction factor $r_T = 0.85$, the search



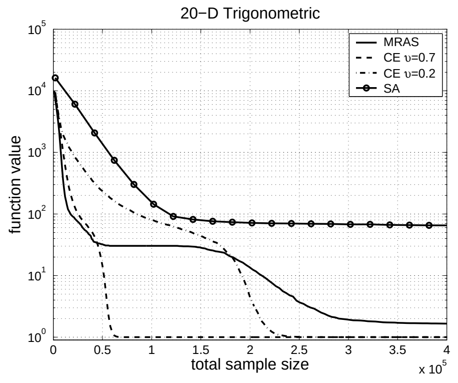
(a) H_6



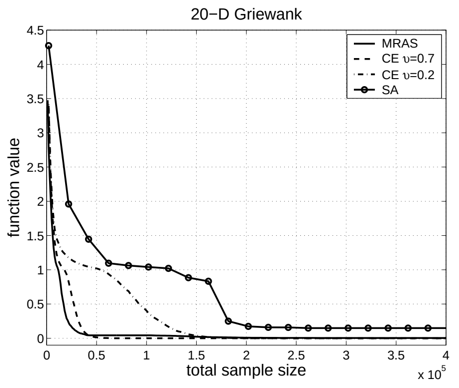
(b) H_7



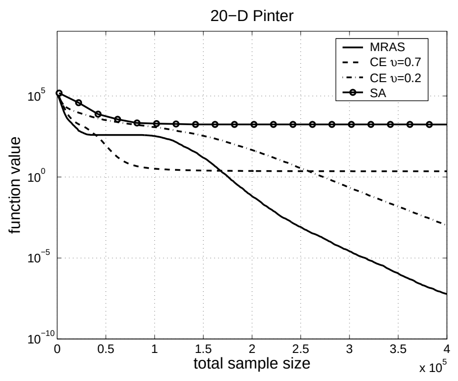
(c) H_8



(d) H_9



(e) H_{10}



(f) H_{11}

Figure 5.5: Average performance (mean of 50 replications) of MRAS, CE, and SA on selected benchmark problems.

neighborhood of a point x is taken to be $\mathcal{N}(x) = \{y : \max_{1 \le i \le n} |x_i - y_i| \le 1\}$, and the initial solution is uniformly selected from $[-50, 50]^n$.

For each problem, we performed 50 independent runs of all three algorithms, and numerical results are reported in Table 5.4, where $\bar{H}_i^*$ is the averaged value of the function $H_i(\cdot)$ at the best solution visited by the algorithm, with standard error in parentheses, and M_ε indicates the number of runs that an ε-optimal solution was found out of 50 trials. We also plotted in Figure 5.5 the average function values of the current best solution given the number of samples generated for selected benchmark problems. The performance comparison is based on the same amount of computational effort, where for each algorithm, the total number of function evaluations (i.e., sample size) is set to $100,000$ for H_6, and $400,000$ for $H_7 - H_{11}$. Here, we choose to use the total number of function evaluations to estimate the computational efforts of different algorithms, because the running time of all three algorithms is dominated by the time spent in evaluating the objective function.

Functions H_6 has only a few local minima, and since SA combines local search, it may quickly locate one of them. However, as we can see, SA stops making improvement during the early search phase. This is caused by the plateaus surrounding the local minima, which makes it very difficult for SA to escape local optima. In contrast, since both MRAS$_1$ and CE are population-based, they show more robustness in dealing with local optima. We see that CE ($\upsilon = 0.7$) does not always converge to the global optimal solution, but it still performs better than SA does. Note that decreasing the value of the smoothing parameter slows down the convergence of CE. In particular, for the $\upsilon = 0.2$ case, although better average function values are achieved in CE, no ε-optimal solutions were found within the allowed simulation budget because of the slow convergence. MRAS$_1$

consistently finds ε-optimal solutions in all simulation runs.

For H_7, none of these three algorithms found ε-optimal solutions. However, Figure 5.5(b) indicates that both MRAS$_1$ and CE perform better than SA when the total sample size is large enough. CE with $\upsilon = 0.2$ converges slowly, but slightly outperforms CE ($\upsilon = 0.7$) after about $170,000$ function evaluations. MRAS$_1$ performs the best, it has a similar convergence rate as CE ($\upsilon = 0.7$) and finds better solutions than the other algorithms do. On H_8, MRAS$_1$ is clearly superior to both CE and SA. It converges to the global optimal solution in all 50 runs at an exponential rate. The performance of SA is similar to the H_7 case, whereas the performance of CE ($\upsilon = 0.7$) is even worse than that of SA, as we can see, the algorithm frequently gets trapped at solutions that are far from optimal. CE with $\upsilon = 0.2$ yields much better performance.

H_9 and H_{10} are highly multimodal functions. CE ($\upsilon = 0.7$) works better than both MRAS$_1$ and SA. It not only converges the fastest but also finds ε-optimal solutions in almost all runs. SA finds no ε-optimal solutions in any of the runs. MRAS$_1$ consistently outperforms SA, and converges to the optimal solution in 50% of the total simulation runs in both cases. Initially, MRAS$_1$ converges very fast to good values near the optimum, then it proceeds at a slower rate and spends most of the time in fine-tuning the solution. The behavior of MRAS$_1$ can be explained by looking at the parameter updating equations (5.16) and (5.17). Since the values of H_9 and H_{10} at local minima near the optimum are very close to each other, the parameter updating in MRAS$_1$ is dominated by the density function in the denominator, especially when the iteration counter k is small.

H_{11} contains both a badly-scaled quadratic term and some badly-scaled noise terms. For this function, SA does not seem to be competitive at all. Similar to the H_7 and H_8 cases, CE ($\upsilon = 0.7$) converges the fastest, but stagnates at some non-optimal solutions in all runs. Using $v = 0.2$ in CE greatly improves the solution quality but slows down the convergence speed. The initial behavior of MRAS$_1$ is similar to the H_9 and H_{10} cases, but the algorithm outperforms CE ($v = 0.7$) after about $170,000$ function evaluations, and then approaches the optimum at an exponential rate.

The above comparison seems to suggest that MRAS$_1$ is better adapted to optimization of badly scaled multimodal problems, whereas CE works best on problems that are well-scaled and contain a large number of local optima. Of course, a more comprehensive numerical study needs to be carried out in order to confirm this finding.

5.6.2 Combinatorial Optimization

In this Chapter, we present the performance of MRAS on various ATSP problems. All test cases are taken from the URL

`http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95.`

For each ATSP problem with N_c cities, an N_c-by-N_c distance matrix G is given, whose (i,j)th element $G_{i,j}$ represents the distance from city i to city j. The goal is to find the shortest path that visits all the cities and returns to the starting city. Mathematically, the problem can be formulated as follows:

$$\min_{x\in\mathcal{X}} H(x) := \min_{x\in\mathcal{X}} \left\{ \sum_{i=1}^{N_c-1} G_{x_i,x_{i+1}} + G_{x_{N_c},x_1} \right\}, \tag{5.33}$$

where $x := (x_1, x_2, \ldots, x_{N_c}, x_1)$ is an admissible tour, and $\mathcal{X}$ is the set of all admissible tours.

We use the same technique as in [67] and [26] for solving these problems, i.e., we associate for each distance matrix G an initial state transition matrix $\widetilde{P}_0$, whose $(i,j)th$ element specifies the probability of transitioning from city i to city j. Thus, at each iteration of MRAS the following two steps are fundamental:

- Generating random (admissible) tours according to the transition matrix and evaluate the performance of each sample tour.

- Updating the transition matrix based on the sample tours generated from the previous step.

The detailed discussion of how to generate admissible tours can be found in e.g., [26]. We now briefly address the issue of how to update the transition matrix. At each iteration k of MRAS, the p.d.f. $f(\cdot, \widetilde{P}_k)$ on $\mathcal{X}$ is parameterized by the transition matrix $\widetilde{P}_k$ and is given by

$$f(x, \widetilde{P}_k) = \prod_{l=1}^{N_c} \sum_{i,j}^{N_c} \widetilde{P}_k(i,j) I_{\{x \in \mathcal{X}_{i,j}(l)\}},$$

where $\mathcal{X}_{i,j}(l)$ is the set of all tours in $\mathcal{X}$ such that the lth transition is from city i to city j. It is straightforward to show that the new transition matrix $\widetilde{P}_{k+1}$ is updated in equation (5.21) as

$$\widetilde{P}_{k+1}(i,j) = \frac{\sum_{l=1}^{N_k} \widetilde{S}_k(H(X_l^k)) I_{\{H(X_l^k) \le \bar{\gamma}_{k+1}\}} I_{\{X_l^k \in \mathcal{X}_{i,j}\}}}{\sum_{l=1}^{N_k} \widetilde{S}_k(H(X_l^k)) I_{\{H(X_l^k) \le \bar{\gamma}_{k+1}\}}}, \tag{5.34}$$

where $X_1^k, \ldots, X_{N_k}^k$ are the i.i.d. sample tours generated from $\widetilde{f}(\cdot, \widetilde{P}_k)$, $\bar{\gamma}_{k+1}$ is defined as in equation (5.22), and $\mathcal{X}_{i,j}$ represents the set of tours in which the transition from city i to city j is made.

The performance of the algorithm on various ATSP problems is reported in Table 5.5. For each of the 7 instances, we performed 10 independent runs of the algorithm. In Table 5.5, N_{total} is the total number of tours generated (mean and standard error reported), H_{best} is the length of the shortest path, H_* and H^* are the worst and best solutions obtained out of 10 trials, δ_* and δ^* are the respective relative errors for H_* and H^*, and δ is the relative error (mean and standard error reported). For all cases, $\varepsilon = 1$, the initial samples $N_0 = 1000$, $\rho_0 = 0.1$, $\lambda = 0.02$, $\alpha = 1.5$, $r = 0.1$, the stopping control parameters $d = 5,\ \tau = 0,\ N_{max} = 10N_c^2$, smoothing parameter $\upsilon = 0.5$, and the initial transition matrix $\widetilde{P}_0$ is initialized as a stochastic matrix whose $(i, j)th$ entry is proportional to the inverse of the (i, j)th entry of G, i.e., $\widetilde{P}_0(i, j) \propto \frac{1}{G_{i,j}}$ and $\sum_j \widetilde{P}_0(i, j) = 1\ \forall i$.

ATSP	N_c	N_{total} (std err)	H_{best}	H_*	H^*	δ_*	δ^*	δ (std err)
ftv33	34	7.95e+4(3.25e+3)	1286	1364	1286	0.061	0.000	0.023(0.008)
ftv35	36	1.02e+5(3.08e+3)	1473	1500	1475	0.018	0.001	0.008(0.002)
ftv38	39	1.31e+5(4.90e+3)	1530	1563	1530	0.022	0.000	0.008(0.003)
p43	43	1.02e+5(4.67e+3)	5620	5637	5620	0.003	0.000	0.001(2.5e-4)
ry48p	48	2.62e+5(1.59e+4)	14422	14810	14446	0.027	0.002	0.012(0.003)
ft53	53	2.94e+5(1.58e+4)	6905	7236	6973	0.048	0.010	0.029(0.005)
ft70	70	4.73e+5(2.91e+4)	38673	39751	38744	0.028	0.002	0.017(0.003)

Table 5.5: Performance of MRAS on various ATSP problems based on 10 independent replications. The standard errors are in parentheses.

5.7 Conclusions

In this Chapter, we have proposed a randomized search technique called Model Reference Adaptive Search (MRAS) for solving general global optimization problems. The method iteratively updates a parameterized probability distribution over the solution space so that the sequence of candidate solutions generated from this distribution will converge asymptotically to the global optimum. We have provided a particular instantiation of the framework and established its global convergence properties in both continuous and discrete (combinatorial) domains. In addition, we have explored the relationship between the recently proposed Cross-Entropy (CE) method and MRAS, and showed that the CE method can also be interpreted as an instance of the MRAS framework. Finally, we have also carried out detailed numerical experiments to investigate the performance of the method.

Throughout this whole chapter, most of the theoretical and empirical analysis work has been focused on an instantiation of the framework. However, we emphasize that the contribution of this research goes far beyond this particular instantiation in that it provides a general framework for designing and analyzing various model-based optimization algorithms. In MRAS, the task of sampling candidate solutions and the task of updating probabilistic models are split in a natural way, and there is considerable flexibility in the choices of reference distributions. Thus, by carefully selecting the reference distributions, one can construct different instantiations of the framework. Moreover, the convergence analysis of these instantiation algorithms can simply be ascribed to the study of the properties of the reference distributions.

The MRAS_1 algorithm demonstrated great promise on some preliminary examples, but practical implementation issues remain. For example, selection of the input parameters in our numerical experiments was based mainly on trial and error. For a given problem, how to determine a priori the most appropriate values of these parameters is an open issue. Designing an adaptive scheme to update these parameters during the search process may also enhance the convergence rate of the algorithm.

A more important line of research is to extend the MRAS method to stochastic optimization problems, where the function values can only be observed in the presence of noise. The construction of a practically efficient generalization of MRAS with provable convergence is addressed in Chapter 6.

Chapter 6

A Model Reference Adaptive Search Method for Stochastic Global Optimization

6.1 Introduction and Motivation

In Chapter 5, we have proposed a unifying framework called Model Reference Adaptive Search (MRAS) for solving *deterministic* global optimization problems. In this Chapter, we discuss how to extend the framework to solving stochastic optimization problems. Stochastic problems arise in a wide range of areas such as manufacturing, communication networks, system design, and financial engineering. In contrast to their deterministic counterparts, such problems are typically much more difficult to solve, either because an explicit relation between the objective function and the underlying decision variables is unavailable or because the cost of a precise evaluation of the objective function is too prohibitive. Oftentimes, one has to use simulation or real-time observations to evaluate the objective function. In such situations, all the objective function evaluations will contain some noise, so special techniques are generally used (as opposed to the deterministic optimization methods) in order to filter out the noisy components.

There are two major techniques to address the function evaluation noise arising from the stochastic setting. One simple approach is to spend a significant amount of computational effort at each point the algorithm visits in order to obtain a precise estimate of the objective function value, and then use deterministic optimization approach to solve the underlying problem. In this respect, the extension of MRAS to stochastic settings should be relatively straightforward. However, questions arise as to what really quantifies a precise estimate, how much computational effort should be invested at each point, and how the estimates of the objective function values will affect the final solutions of the algorithm. These questions are not easy to answer; moreover, when the function evaluation cost is expensive, to obtain a precise estimate of the objective function value is often infeasible. To circumvent these difficulties, we resort to the alternative approach, which does not require obtaining highly precise estimates of the objective function values each time the algorithm visits a solution. However, we need to modify the MRAS approach intended for deterministic problems in order to yield good performance in the presence of noise.

The method we propose in this Chapter is called stochastic model reference adaptive search (SMRAS), which is essentially a generalization of the MRAS method for deterministic optimization with some appropriate modifications and extensions required for the stochastic setting. The idea behind SMRAS, as in MRAS for deterministic optimization, is to use a pre-specified parameterized probability distribution family to generate candidate solutions, and to use a sequence of convergent *reference* distributions to facilitate and guide the updating of the parameters associated with the parameterized family at each step of the iteration procedure. A major modification from the original MRAS method is in the way the sequence of *reference* distributions is constructed. In MRAS, *reference* distributions are *idealized* probabilistic models constructed based on the exact performance of the candidate solutions. In the stochastic case, however, the objective function cannot be evaluated deterministically, so the sample average approximations of the (idealized) *reference* distributions are used in SMRAS to guide the parameter updating. We show that for a class of parameterized distributions, i.e., the so-called Natural Exponential Family (NEF), SMRAS converges with probability one to a global optimal solution for both stochastic continuous and discrete problems. To the best of our knowledge, SMRAS is the first *model-based* search method for solving *general* stochastic optimization problems with provable convergence.

6.2 A Brief Review of Stochastic Optimization Solution Techniques

There are some obvious distinctions between the solution techniques for stochastic optimization when the decision variable is continuous and when it is discrete. Although some techniques, in principle, can be applied to both types of problems, they require some suitable modifications in order to switch from one setting to another.

A well-known class of methods for solving stochastic optimization problems with continuous decision variables is stochastic approximation (SA). These methods mimic the classical gradient-based search method in deterministic optimization, and rely on the estimation of the gradient of the objective function with respect to the decision variables. Because they are gradient-based, these methods generally find local optimal solutions. In terms of the different gradient estimation techniques employed, the SA algorithms can be generally divided into two categories: algorithms that are based on direct gradient estimation techniques, the best-known of which are perturbation analysis (PA) and the likelihood ratio/score function (LR/SF) method ([69]), and algorithms that are based on indirect gradient estimation techniques like finite difference and its variations ([78]). A detailed review of various gradient estimation techniques can be found in [32].

When the underlying decision variables are discrete, one popular approach is to use random search. This has given rise to many different stochastic discrete optimization algorithms, including the stochastic ruler method and its modification ([4], [87]), the random search methods ([6], [7]), modified simulated annealing ([5]), and the nested partitions method of [76]. The main idea throughout is to construct a Markov chain over the solution space and show that the Markov chain settles down on the set of (possibly local) optimal solutions.

From an algorithmic point of view (cf. Chapter 1.2), the aforementioned approaches are *instance-based* techniques. There are also some independently developed *model-based* methods that can also be applied to stochastic *discrete* optimization problems. Two most well-established methods are the Stochastic Ant Colony Optimization (S-ACO) ([37]) and the Cross-Entropy (CE) method (cf. e.g., [26], [65], [66], [67], [68]) The S-ACO method is the extension of the original Ant Colony Optimization (ACO) algorithm ([29]) to stochastic problems. The method uses Monte-Carlo sampling to estimate the objective and is shown (under some regularity assumptions) to converge to the global optimal solution for the stochastic combinatorial problems with probability one. The CE method was motivated by an adaptive algorithm for estimating probabilities of rare events. It was later realized that the method can be modified to solve deterministic optimization problems (cf. e.g., [66]). More recently, Rubinstein [67] shows that the method is also capable of handling the stochastic network combinatorial optimization problems, and in that context, establishes the probability one convergence of the algorithm.

6.3 The Stochastic Model Reference Adaptive Search Method

We consider the following optimization problem:

$$x^* \in \arg\max_{x \in \mathcal{X}} E_\psi[H(x, \psi)], \quad x \in \mathcal{X} \subseteq \Re^n, \tag{6.1}$$

where $\mathcal{X}$ is the solution space, which can be either continuous or discrete, $H(\cdot, \cdot)$ is a deterministic, real-valued function, and ψ is a random variable (possibly depending on x) representing the stochastic effects of the system. We let $h(x) := E_\psi[H(x, \psi)]$, and assume that $h(x)$ cannot be obtained easily, but the random variable $H(x, \psi)$ can be observed, e.g., via simulation. We assume throughout that (6.1) has a unique global optimal solution, i.e., $\exists\, x^* \in \mathcal{X}$ such that $h(x) < h(x^*)\ \forall\, x \neq x^*,\ x \in \mathcal{X}$. We also assume that random samplings can be done easily on $\mathcal{X}$, at least for a class of distributions of interest.

6.3.1 General Framework

Similar to MRAS, SMRAS uses a family of parameterized distributions $\{f(\cdot,\theta),\ \theta \in \Theta\}$ as sampling distribution to generate candidate solutions, where Θ is some parameter space. The basic algorithmic structure is very simple. At each iteration k, suppose we have already obtained a parameter θ_k, then the main body of the method consists of the following two steps:

1. Generate candidate solutions from the current sampling distribution $f(\cdot,\theta_k)$.
2. Compute a new parameter vector θ_{k+1} according to a specified parameter updating rule by using the candidate solutions generated in the previous step in order to concentrate the future search toward more promising regions.

The parameter updating rule in SMRAS is guided by another sequence of distributions $\{\widetilde{g}_k(\cdot)\}$, called the *reference* distribution. These reference distributions are used to express the desired properties of the method; thus we may often want to construct them such that they will have some nice theoretical properties (however, they could be difficult to handle in practice). Once these reference distributions are specified, then at each iteration k, we look at the *projection* of $\widetilde{g}_k(\cdot)$ on the family of distributions $\{f(\cdot,\theta)\}$ and compute the new parameter vector θ_{k+1} that minimizes the Kullback-Leibler (KL) distance

$$\mathcal{D}(\widetilde{g}_k, f(\cdot,\theta)) := E_{\widetilde{g}_k}\left[\ln \frac{\widetilde{g}_k(X)}{f(X,\theta)}\right] = \int_{\mathcal{X}} \ln \frac{\widetilde{g}_k(x)}{f(x,\theta)} \widetilde{g}_k(dx),$$

where $X = (X_1, \ldots, X_n)$ is a random vector having distribution $\widetilde{g}_k(\cdot)$ and taking values in $\mathcal{X}$, and $E_{\widetilde{g}_k}[\cdot]$ represents the expectation taken with respect to $\widetilde{g}_k(\cdot)$. Intuitively speaking, under the KL-distance measure, $f(\cdot, \theta_{k+1})$ can be viewed as a compact approximation of the reference distribution and thus may share some similar properties with $\widetilde{g}_k(\cdot)$. Therefore, to ensure the convergence of SMRAS, one basic property the sequence $\{\widetilde{g}_k(\cdot)\}$ should have is convergence. There could be many different ways to construct such a convergent sequence of distributions. When the performance measure is deterministic, we have proposed in Chapter 5.2 the following simple iterative method for constructing the reference distribution $\{g_k(\cdot)\}$. Let $g_0(x) > 0,\ \forall x \in \mathcal{X}$ be an initial probability density/mass function (p.d.f./p.m.f.) on the solution space $\mathcal{X}$. Then, at each iteration $k \geq 1$, compute a new p.d.f./p.m.f. by tilting the old p.d.f./p.m.f. $g_{k-1}(x)$ with the performance function $h(x)$ (for simplicity, here we assume that $h(x) > 0,\ \forall x \in \mathcal{X}$), i.e.,

$$g_k(x) = \frac{h(x) g_{k-1}(x)}{\int_{\mathcal{X}} h(x) g_{k-1}(dx)}, \quad \forall x \in \mathcal{X}. \tag{6.2}$$

It is possible to show that the sequence of p.d.f.'s $\{g_k(\cdot)\}$ constructed above will converge to a p.d.f. that concentrates only on the set of optimal solutions, regardless of the initial $g_0(\cdot)$ used. However, in the stochastic setting, since the performance function $h(\cdot)$ cannot be evaluated exactly, the iteration procedure given by (6.2) is no longer applicable. Thus, in SMRAS, one key modification from the original deterministic approach is to use approximations $\{\widetilde{g}_k(\cdot)\}$ of $\{g_k(\cdot)\}$ as the sequence of reference distributions, which are constructed based on the sample average approximation of the expected performance function $h(\cdot)$.

6.3.2 Algorithm Description

In SMRAS, there are two allocation rules. The first one, denoted by $\{N_k, k = 0, 1 \ldots\}$, is called the sampling allocation rule, where each N_k determines the number of candidate solutions to be generated from the current sampling distribution at the kth iteration. The second is the observation allocation rule $\{M_k, k = 0, 1, \ldots\}$, which allocates M_k simulation observations to each of the candidate solutions generated at the kth iteration. We require both N_k and M_k to increase as the number of iteration grows for convergence, but other than that, there is considerable flexibility in their choices. To fix ideas, we use a parameter $\alpha > 1$, specified initially, to control the rate of increase in $\{N_k, k = 0, 1 \ldots\}$, and leave the sequence $\{M_k, k = 0, 1, \ldots\}$ as user-specified. When M_k observations are allocated to a solution x at iteration k, we use $H_j(x)$ to denote the jth (independent) random observation of $H(x, \psi)$, and use $\bar{H}_k(x) = \frac{1}{M_k}\sum_{j=1}^{M_k} H_j(x)$ to denote the sample average of all M_k observations made at x.

The performance of the SMRAS algorithm depends on another important sequence of quantities $\{\rho_k, k = 0, 1 \ldots\}$. The motivation behind the sequence is to distinguish "good" samples from "bad" ones and to concentrate the computational effort on the set of promising samples. The sequence $\{\rho_k\}$ is fully adaptive and works cooperatively with the sequence $\{N_k\}$. At successive iterations of the algorithm, a sequence of thresholds $\{\bar{\gamma}_k, k = 1, 2, \ldots\}$ is generated according to the sequence of sample $(1 - \rho_k)$-quantiles, and only those samples that have performances better than these thresholds will be used in parameter updating. Thus, each ρ_k determines the approximate proportion of N_k samples that will be used to update the probabilistic model at iteration k.

During the initialization step of SMRAS, a small positive number ε and a continuous and strictly increasing function $S(\cdot) : \Re \to \Re^+$ are specified. The role of the parameter ε, as we will see later, is to filter out the observation noise. The function $S(\cdot)$ is used to account for the cases where the sample average approximations $\bar{H}_k(x)$ are negative for some x.

At each iteration k, random samples are drawn from the density/mass function

```
**Stochastic Model Reference Adaptive Search (SMRAS)**

• **Initialization:** Specify $\rho_0 \in (0,1]$, $N_0 > 1$, $\alpha > 1$, $\varepsilon > 0$, an allocation rule $\{M_k\}$, a strictly
  increasing $S(\cdot) : \Re \to \Re^+$, mixing coefficients $\{\lambda_k,\ k = 0,1,\ldots\}$ satisfying $\lambda_k \geq \lambda_{k+1}$ and $\lambda_k \in$
  $(0,1)\ \forall\, k$, and an initial p.d.f. $f(x,\theta_0) > 0\ \forall\, x \in \mathcal{X}$. Set $k \leftarrow 0$.

• **Repeat until a specified stopping rule is satisfied:**

    1. Generate $N_k$ samples $X_1^k, \ldots, X_{N_k}^k$ according to $\widetilde{f}(\cdot,\theta_k) := (1-\lambda_k) f(\cdot,\theta_k) + \lambda_k f(\cdot,\theta_0)$.

    2. Compute the sample $(1-\rho_k)$-quantile $\widetilde{\gamma}_{k+1}(\rho_k, N_k) := \bar{H}_{k,(\lceil (1-\rho_k) N_k \rceil)}$, where $\lceil a \rceil$ is the
       smallest integer greater than $a$, and $\bar{H}_{k,(i)}$ is the $i$th order statistic of the sequence
       $\{\bar{H}_k(X_i^k),\ i = 1,\ldots,N_k\}$.

    3. **If** $k = 0$ **or** $\widetilde{\gamma}_{k+1}(\rho_k, N_k) \geq \bar{\gamma}_k + \varepsilon$, **then** do step 3a.

           3a. Set $\bar{\gamma}_{k+1} \leftarrow \widetilde{\gamma}_{k+1}(\rho_k, N_k),\ \rho_{k+1} \leftarrow \rho_k,\ N_{k+1} \leftarrow N_k,\ X_{k+1}^{\dagger} \leftarrow X_{1-\rho_k}$, where
               $X_{1-\rho_{k+1}} \in \{x : \bar{H}_k(x) = \bar{H}_{k,(\lceil (1-\rho_{k+1}) N_k \rceil)},\ x \in \{X_1^k, \ldots, X_{N_k}^k\}\}$.

       **else**, find the largest $\bar{\rho} \in (0, \rho_k)$ such that $\widetilde{\gamma}_{k+1}(\bar{\rho}, N_k) \geq \bar{\gamma}_k + \varepsilon$.

           3b. **If** $\bar{\rho}$ exists, **then** set $\bar{\gamma}_{k+1} \leftarrow \widetilde{\gamma}_{k+1}(\bar{\rho}, N_k),\ \rho_{k+1} \leftarrow \bar{\rho},\ N_{k+1} \leftarrow N_k,\ X_{k+1}^{\dagger} \leftarrow X_{1-\bar{\rho}}$.

           3c. **else** if no $\bar{\rho}$ exists, set $\bar{\gamma}_{k+1} \leftarrow \bar{H}_k(X_k^{\dagger}),\ \rho_{k+1} \leftarrow \rho_k,\ N_{k+1} \leftarrow \lceil \alpha N_k \rceil,\ X_{k+1}^{\dagger} \leftarrow X_k^{\dagger}$.

       **endif**

    4. Compute $\theta_{k+1}$ as

       $$\theta_{k+1} = \arg\max_{\theta \in \Theta} \frac{1}{N_k} \sum_{i=1}^{N_k} \frac{[S(\bar{H}_k(X_i^k))]^k}{\widetilde{f}(X_i^k, \theta_k)} \widetilde{I}[\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}] \ln f(X_i^k, \theta), \qquad (6.3)$$

       where $\widetilde{I}(x,\gamma) := \begin{cases} 1 & \text{if } x \geq \gamma, \\ (x - \gamma + \varepsilon)/\varepsilon & \text{if } \gamma - \varepsilon < x < \gamma, \\ 0 & \text{if } x \leq \gamma - \varepsilon. \end{cases}$

    5. Set $k \leftarrow k + 1$.
```

Figure 6.1: Stochastic Model Reference Adaptive Search

$\widetilde{f}(\cdot, \theta_k)$, which is a mixture of the initial density $f(\cdot, \theta_0)$ and the density calculated from the previous iteration $f(\cdot, \theta_k)$. The initial density $f(\cdot, \theta_0)$ can be chosen according to some prior knowledge of the problem structure; however, if nothing is known about where the good solutions are, this density should be chosen in such a way that each region in the solution space will have an (approximately) equal probability of being sampled. Intuitively, mixing in the initial density enables the algorithm to explore the entire solution space and thus maintain a global perspective during the search process.

At step 2, the sample $(1 - \rho_k)$-quantile $\widetilde{\gamma}_{k+1}$ with respect to $\widetilde{f}(\cdot, \theta_k)$ is calculated by first ordering the sample performances $\bar{H}_k(X_i^k),\ i = 1, \ldots, N_k$ from smallest to largest, $\bar{H}_{k,(1)} \leq \bar{H}_{k,(2)} \leq \cdots \leq \bar{H}_{k,(N_k)}$, and then taking the $\lceil (1 - \rho_k) N_k \rceil$th order statistic. We use the function $\widetilde{\gamma}_{k+1}(\rho_k, N_k)$ to emphasize the dependencies of $\widetilde{\gamma}_{k+1}$ on both ρ_k and N_k, so that different sample quantile values can be distinguished by their arguments.

Step 3 of the algorithm is used to construct a sequence of thresholds $\{\bar{\gamma}_k, k = 1, 2, \ldots\}$ from the sequence of sample quantiles $\{\widetilde{\gamma}_k\}$, and to determine the appropriate values of the ρ_{k+1} and N_{k+1} to be used in subsequent iterations. This is carried out by checking whether the condition $\widetilde{\gamma}_{k+1}(\rho_k, N_k) \geq \bar{\gamma}_k + \varepsilon$ is satisfied. If the inequality holds, then both the current ρ_k value and the new sample size N_k are satisfactory, and $\widetilde{\gamma}_{k+1}(\rho_k, N_k)$ is used as the current threshold value. Otherwise, we fix the sample size N_k and try to find a smaller $\bar{\rho} < \rho_k$ such that the above inequality can be satisfied with the new sample $(1 - \bar{\rho})$-quantile. If such a $\bar{\rho}$ does exist, then the current sample size N_k is still deemed acceptable, and the new threshold value is updated by the sample $(1 - \bar{\rho})$-quantile. On the other hand, if no such $\bar{\rho}$ can be found, then the sample size N_k is increased by a factor α, and the new threshold $\bar{\gamma}_{k+1}$ is calculated by using an additional variable $X_k^\dagger$ to remember the particular sample that achieves the previous threshold value $\bar{\gamma}_k$, and then simply allocating M_k observations to $X_k^\dagger$. It is important to note that in step 4, the set $\{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon, x \in \{X_1^k, \ldots, X_{N_k}^k\}\}$ could be empty, since it could happen that all the random samples generated at the current iteration are much worse than those generated at the previous iteration. If this is the case, then by the definition of $\widetilde{I}(\cdot,\cdot,)$, the right hand side of equation (6.3) will be equal to zero, so any $\theta \in \Theta$ is a maximizer; we define $\theta_{k+1} := \theta_k$ in this case. Note that a "soft" threshold function $\widetilde{I}(\cdot,\cdot)$, as opposed to the indicator function, is used in parameter updating (cf. equations (6.3)). The reason for doing so, as will be explained later, is to smooth out the noisy observations.

We now show that there is a sequence of *reference* models $\{\widetilde{g}_k(\cdot)\}$ implicit in SMRAS, and the parameter θ_{k+1} computed at step 4 indeed minimizes the KL-divergence $\mathcal{D}(\widetilde{g}_{k+1}, f(\cdot,\theta))$.

Lemma 6.3.1 *The parameter θ_{k+1} computed at the kth iteration of SMRAS minimizes the KL-divergence $\mathcal{D}\left(\widetilde{g}_{k+1}, f(\cdot,\theta)\right)$, where*

$$\widetilde{g}_{k+1}(x) := \begin{cases} \dfrac{\left[[S(\bar{H}_k(x))]^k / \widetilde{f}(x,\theta_k)\right] \widetilde{I}\left(\bar{H}_k(x), \bar{\gamma}_{k+1}\right)}{\sum_{i=1}^{N_k} \left[[S(\bar{H}_k(X_i^k))]^k / \widetilde{f}(X_i^k,\theta_k)\right] \widetilde{I}\left(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}\right)} & \text{if } \left\{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon,\ x \in \Lambda_k \neq \emptyset\right\}, \\ \widetilde{g}_k(x) & \text{otherwise}, \end{cases} \tag{6.4}$$

$$\forall\, k = 0, 1, \ldots, \text{ where } \bar{\gamma}_{k+1} := \begin{cases} \widetilde{\gamma}_{k+1}(\rho_k, N_k) & \text{if step 3a is visited}, \\ \widetilde{\gamma}_{k+1}(\bar{\rho}, N_k) & \text{if step 3b is visited}, \\ \bar{H}_k(X_k^\dagger) & \text{if step 3c is visited}, \end{cases} \text{ and } \Lambda_k := \{X_1^k, \ldots, X_{N_k}^k\}.$$

Proof: We only need to consider the case where $\left\{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon,\ x \in \Lambda_k\right\} \neq \emptyset$, since if this is not the case, then we can always backtrack and find a $\widetilde{g}_k(\cdot)$ with non-empty support.

For brevity, we define $\widetilde{S}_k(\bar{H}_k(x)) := \frac{[S(\bar{H}_k(x))]^k}{\widetilde{f}(x,\theta_k)}$. Note that at the kth iteration, the

K-L divergence between $\widetilde{g}_{k+1}(\cdot)$ and $f(\cdot,\theta)$ can be written as

$$\begin{aligned}
&\mathcal{D}\left(\widetilde{g}_{k+1}, f(\cdot,\theta)\right) \\
=\ & E_{\widetilde{g}_{k+1}}\left[\ln \widetilde{g}_{k+1}(X)\right] - E_{\widetilde{g}_{k+1}}\left[\ln f(X,\theta)\right] \\
=\ & E_{\widetilde{g}_{k+1}}\left[\ln \widetilde{g}_{k+1}(X)\right] - \frac{\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\left(\bar{H}_k(X_i^k),\bar{\gamma}_{k+1}\right)\ln f(X_i^k,\theta)}{\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\left(\bar{H}_k(X_i^k),\bar{\gamma}_{k+1}\right)},
\end{aligned}$$

where X is a random variable with distribution $\widetilde{g}_{k+1}(\cdot)$. Thus the proof is completed by observing that minimizing $\mathcal{D}\left(\widetilde{g}_{k+1}, f(\cdot,\theta)\right)$ is equivalent to maximizing the quantity $\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\left(\bar{H}_k(X_i^k),\bar{\gamma}_{k+1}\right)\ln f(X_i^k,\theta)$. ∎

Remark 6.3.1 *For optimization problems with finite solution spaces, it is often useful to make efficient use of the past sampling information. This can be achieved by maintaining a list of all sampled candidate solutions as well as the number of observations made at each of these visited solutions, and then check if a newly generated solution is in that list. If at the kth iteration, a new solution has already been visited and, say M_l, observations have been allocated, then we only need to take $M_k - M_l$ additional observations from that point. This procedure is often effective when the solution space is relatively small. However, when the solution space is large, the storage and checking cost could be quite expensive. In SMRAS, we propose an alternative approach: at each iteration k of the method, instead of remembering all past samples, we only keep track of those samples that fall in the region $\left\{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon\right\}$. Thus, as the search becomes more and more concentrated on these regions, the probability of getting repeated samples will typically increase.*

6.4 Convergence Analysis

For reasons discussed in Chapter 5, we restrict our discussion to the so-called natural exponential family (NEF) (see Definition 5.3.1), which works well in practice, and for which convergence properties can be established.

We make the following assumptions about the noisy observations $H_j(x)$ and the observation allocation rule $\{M_k\}$.

Assumptions:

$L1$. *For any given $\varepsilon > 0$, these exists a positive number n^* such that for all $n \geq n^*$,*

$$\sup_{x\in\mathcal{X}} P\Big(\Big|\frac{1}{n}\sum_{j=1}^{n} H_j(x) - h(x)\Big| \geq \varepsilon\Big) \leq \phi(n,\varepsilon),$$

where $\phi(\cdot,\cdot)$ is strictly decreasing in its first argument and non-increasing in its second argument. Moreover, $\phi\,(n,\varepsilon) \to 0$ as $n \to \infty$.

$L2$. *For any $\varepsilon > 0$, there exist positive numbers m^* and n^* such that for all $m \geq m^*$ and $n \geq n^*$,*

$$\sup_{x,y\in\mathcal{X}} P\Big(\Big|\frac{1}{m}\sum_{j=1}^{m} H_j(x) - \frac{1}{n}\sum_{j=1}^{n} H_j(y) - h(x) + h(y)\Big| \geq \varepsilon\Big) \leq \phi\left(\min\{m,n\},\varepsilon\right),$$

where $\phi(\cdot,\cdot)$ satisfies the conditions in L1.

$L3$. *The observation allocation rule $\{M_k, k = 0,1,\ldots\}$ satisfies $M_k \geq M_{k-1}$ $\forall$ $k = 1,2,\ldots,$ and $M_k \to \infty$ as $k \to \infty$. Moreover, for any $\varepsilon > 0$, there exist $\delta_\varepsilon \in (0,1)$ and $\mathcal{K}_\varepsilon > 0$ such that $\alpha^{2k}\phi(M_{k-1},\varepsilon) \leq (\delta_\varepsilon)^k$, $\forall$ $k \geq \mathcal{K}_\varepsilon$, where $\phi(\cdot,\cdot)$ is defined as in $L1$.*

Assumption $L1$ is satisfied by many random sequences, e.g., the sequence of i.i.d. random variables with (asymptotically) uniformly bounded variance, or a class of random variables (not necessarily i.i.d.) that satisfy the large deviations principle; please refer to [42] for further details. Assumption $L2$ can be viewed as a simple extension of $L1$. Most random sequences that satisfy $L1$ will also satisfy $L2$. For example, consider the particular case where the sequence $H_j(x), j = 1,2,\ldots$ is i.i.d. with uniformly bounded variance $\sigma^2(x)$ and $E(H_j(x)) = h(x),\ \forall\, x \in \mathcal{X}$. Thus the variance of the random variable $\frac{1}{m}\sum_{j=1}^{m} H_j(x) - \frac{1}{n}\sum_{j=1}^{n} H_j(y)$ is $\frac{1}{m}\sigma^2(x) + \frac{1}{n}\sigma^2(y)$, which is also uniformly bounded on $\mathcal{X}$. By Chebyshev's inequality, we have for any $x, y \in \mathcal{X}$

$$\begin{aligned} P\Big(\Big|\frac{1}{m}\sum_{j=1}^{m} H_j(x) - \frac{1}{n}\sum_{j=1}^{n} H_j(y) - h(x) + h(y)\Big| \geq \varepsilon\Big) &\leq \frac{\sup_{x,y}\left[\frac{1}{m}\sigma^2(x) + \frac{1}{n}\sigma^2(y)\right]}{\varepsilon^2}, \\ &\leq \frac{\sup_{x,y}\left[\sigma^2(x) + \sigma^2(y)\right]}{\min\{m,n\}\varepsilon^2}, \\ &= \phi(\min\{m,n\}, \varepsilon). \end{aligned}$$

Assumption $L3$ is a regularity condition imposed on the observation allocation rule. $L3$ is a mild condition and is very easy to verify. For instance, if $\phi(n, \varepsilon)$ takes the form $\phi(n, \varepsilon) = \frac{\mathcal{C}(\varepsilon)}{n}$, where $\mathcal{C}(\varepsilon)$ is a constant depending on ε, then the condition on M_{k-1} becomes $M_{k-1} \geq \mathcal{C}(\varepsilon)(\frac{\alpha^2}{\delta_\varepsilon})^k\ \forall\, k \geq \mathcal{K}_\varepsilon$. As another example, if $H_j(x), j = 1, 2 \ldots$ satisfies the large deviations principle and $\phi(n, \varepsilon) = e^{-n\mathcal{C}(\varepsilon)}$, then the condition becomes $M_{k-1} \geq \left[\ln(\frac{\alpha^2}{\delta_\varepsilon})/\mathcal{C}(\varepsilon)\right] k,\ \forall\, k \geq \mathcal{K}_\varepsilon$.

To establish the global convergence of SMRAS, we make the following additional assumptions.

Assumptions:

$B1$. *There exists a compact set Π such that for the sequence of random variables $\{X_k^\dagger, k = 1, 2, \ldots\}$ generated by SMRAS, $\exists \mathcal{N} < \infty$ w.p.1 such that $\{x : h(x) \geq h(X_k^\dagger) - \varepsilon\} \cap \mathcal{X} \subseteq \Pi\ \forall\, k \geq \mathcal{N}$.*

$B2$. *For any constant $\xi < h(x^*)$, the set $\{x : h(x) \geq \xi\} \cap \mathcal{X}$ has a strictly positive Lebesgue or discrete measure.*

$B3$. *For any given constant $\delta > 0$, $\sup_{x \in A_\delta} h(x) < h(x^*)$, where $A_\delta := \{x : \|x - x^*\| > \delta\} \cap \mathcal{X}$, and we define the supremum over the empty set to be $-\infty$.*

*B*4. *For each point $z \le h(x^*)$, there exist $\Delta_k > 0$ and $L_k > 0$, such that* $\frac{|(S(z))^k - (S(\bar{z}))^k|}{|(S(z))^k|} \le$ $L_k |z - \bar{z}|$ *for all* $\bar{z} \in (z - \Delta_k, z + \Delta_k)$.

*B*5. *The maximizer of equation (6.3) is an interior point of* Θ *for all* k.

*B*6. $\sup_{\theta \in \Theta} \| \exp\{\theta^T \Gamma(x)\} \Gamma(x) \ell(x) \|$ *is integrable/summable with respect to* x, *where* θ, $\Gamma(\cdot)$, *and* $\ell(\cdot)$ *are defined in Definition 5.3.1.*

*B*7. $f(x, \theta_0) > 0 \; \forall x \in \mathcal{X}$ *and* $f_* := \inf_{x \in \Pi} f(x, \theta_0) > 0$, *where* Π *is defined in* $B1$.

As we will see, the sequence $\{X_k^\dagger\}$ generated by SMRAS converges (cf. the proof of Lemma 6.4.3). Thus, $B1$ requires that the search of SMRAS will eventually end up in a compact set. The assumption is trivially satisfied if the solution space $\mathcal{X}$ is compact. Assumption $B2$ ensures that the neighborhood of the optimal solution x^* will be sampled with a strictly positive probability. Since x^* is the unique global optimizer of $h(\cdot)$, $B3$ is satisfied by many functions encountered in practice. $B4$ can be understood as a locally Lipschitz condition on $[S(\cdot)]^k$; its suitability will be discussed later. In actual implementation of the algorithm, step 4 is often posed as an unconstrained optimization problem, i.e., $\Theta = \Re^m$, in which case $B5$ is automatically satisfied. It is also easy to verify that $B6$ and $B7$ are satisfied by most NEFs.

To show the convergence of SMRAS, we will need the following lemmas.

Lemma 6.4.1 *If Assumptions* $L1$–$L3$ *are satisfied, then step* $3a/3b$ *of SMRAS will be visited finitely often (f.o.) w.p.1 as* $k \to \infty$.

Proof: We consider the sequence $\{X_k^\dagger, k = 1, 2, \ldots\}$ generated by SMRAS, and let $\mathcal{A}_k$ be the event that step $3a/3b$ is visited at the kth iteration, $\mathcal{B}_k := \{h(X_{k+1}^\dagger) - h(X_k^\dagger) \le \frac{\varepsilon}{2}\}$, and $\Lambda_k = \{X_1^k, \ldots, X_{N_k}^k\}$ be the set of candidate solutions generated at the kth iteration. Since the event $\mathcal{A}_k$ implies $\bar{H}_k(X_{k+1}^\dagger) - \bar{H}_{k-1}(X_k^\dagger) \geq \varepsilon$, we have

$$\begin{aligned}
P(\mathcal{A}_k \cap \mathcal{B}_k) &\leq P\Big(\big\{\bar{H}_k(X_{k+1}^\dagger) - \bar{H}_{k-1}(X_k^\dagger) \geq \varepsilon\big\} \cap \big\{h(X_{k+1}^\dagger) - h(X_k^\dagger) \leq \frac{\varepsilon}{2}\big\}\Big) \\
&\leq P\Big(\bigcup_{x\in\Lambda_k, y\in\Lambda_{k-1}} \Big\{\big\{\bar{H}_k(x) - \bar{H}_{k-1}(y) \geq \varepsilon\big\} \cap \big\{h(x) - h(y) \leq \frac{\varepsilon}{2}\big\}\Big\}\Big) \\
&\leq \sum_{x\in\Lambda_k, y\in\Lambda_{k-1}} P\Big(\big\{\bar{H}_k(x) - \bar{H}_{k-1}(y) \geq \varepsilon\big\} \cap \big\{h(x) - h(y) \leq \frac{\varepsilon}{2}\big\}\Big) \\
&\leq |\Lambda_k||\Lambda_{k-1}| \sup_{x,y\in\mathcal{X}} P\Big(\big\{\bar{H}_k(x) - \bar{H}_{k-1}(y) \geq \varepsilon\big\} \cap \big\{h(x) - h(y) \leq \frac{\varepsilon}{2}\big\}\Big) \\
&\leq |\Lambda_k||\Lambda_{k-1}| \sup_{x,y\in\mathcal{X}} P\Big(\bar{H}_k(x) - \bar{H}_{k-1}(y) - h(x) + h(y) \geq \frac{\varepsilon}{2}\Big) \\
&\leq |\Lambda_k||\Lambda_{k-1}|\phi\big(\min\big\{M_k, M_{k-1}\big\}, \frac{\varepsilon}{2}\big) \text{ by Assumption } L2 \\
&\leq \alpha^{2k} N_0^2 \phi\big(M_{k-1}, \frac{\varepsilon}{2}\big) \\
&\leq N_0^2 (\delta_{\varepsilon/2})^k, \ \ \forall\, k \geq \mathcal{K}_{\varepsilon/2} \text{ by Assumption } L3.
\end{aligned}$$

Therefore,

$$\sum_{k=1}^{\infty} P\left(\mathcal{A}_k \cap \mathcal{B}_k\right) \leq \mathcal{K}_{\varepsilon/2} + N_0^2 \sum_{k=\mathcal{K}_{\varepsilon/2}}^{\infty} (\delta_{\varepsilon/2})^k \leq \infty.$$

By the Borel-Cantelli lemma, we have

$$P\left(\mathcal{A}_k \cap \mathcal{B}_k \ i.o.\right) = 0.$$

It follows that if $\mathcal{A}_k$ happens infinitely often, then w.p.1, $\mathcal{B}_k^c$ will also happen infinitely often. Thus,

$$
\begin{aligned}
&\sum_{k=1}^{\infty}\left[h(X^{\dagger}_{k+1})-h(X^{\dagger}_{k})\right]\\
=&\sum_{k:\ \mathcal{A}_k \text{ occurs}}\left[h(X^{\dagger}_{k+1})-h(X^{\dagger}_{k})\right]+\sum_{k:\ \mathcal{A}_k^c \text{ occurs}}\left[h(X^{\dagger}_{k+1})-h(X^{\dagger}_{k})\right]\\
=&\sum_{k:\ \mathcal{A}_k \text{ occurs}}\left[h(X^{\dagger}_{k+1})-h(X^{\dagger}_{k})\right] \text{ since } X^{\dagger}_{k+1}=X^{\dagger}_{k} \text{ if step } 3c \text{ is visited}\\
=&\sum_{k:\ \mathcal{A}_k\cap\mathcal{B}_k \text{ occurs}}\left[h(X^{\dagger}_{k+1})-h(X^{\dagger}_{k})\right]+\sum_{k:\ \mathcal{A}_k\cap\mathcal{B}_k^c \text{ occurs}}\left[h(X^{\dagger}_{k+1})-h(X^{\dagger}_{k})\right]\\
=&\infty \quad w.p.1 \quad \text{since } \varepsilon>0.
\end{aligned}
$$

However, this is a contradiction, since $h(x)$ is bounded from above by $h(x^*)$. Therefore, w.p.1, $\mathcal{A}_k$ can only happen a finite number of times. ∎

Remark 6.4.1 *Lemma 6.4.1 implies that step* 3c *of SMRAS will be visited infinitely often (i.o.) w.p.1.*

Remark 6.4.2 *Note that when the solution space $\mathcal{X}$ is finite, the set Λ_k will be finite for all k. Thus, Lemma 6.4.1 may still hold if we replace Assumption L3 by some milder conditions on M_k. One such condition is $\sum_{k=1}^{\infty}\phi(M_k,\varepsilon)<\infty$, for example, when the sequence $H_j(x), j=1,2\ldots$ satisfies the large deviations principle and $\phi(n,\varepsilon)$ takes the form $\phi(n,\varepsilon)=e^{-n\mathcal{C}(\varepsilon)}$. A particular observation allocation rule that satisfies this condition is $M_k=M_{k-1}+1\ \forall\ k=1,2,\ldots$.*

The following lemma relates the sequence of sampling distributions $\{f(\cdot,\theta_k),k=1,2,\ldots\}$ to the sequence of reference models $\{\widetilde{g}_k(\cdot),k=1,2\ldots\}$ (cf. (6.4)).

Lemma 6.4.2 *If assumptions B5 and B6 hold, then we have*

$$E_{\theta_{k+1}}\left[\Gamma(X)\right]=E_{\widetilde{g}_{k+1}}\left[\Gamma(X)\right],\quad \forall\ k=0,1,\ldots,$$

where $E_{\theta_{k+1}}(\cdot)$ and $E_{\widetilde{g}_{k+1}}(\cdot)$ are the expectations taken with respect to the p.d.f./p.m.f. $f(\cdot,\theta_{k+1})$ and $\widetilde{g}_{k+1}(\cdot)$, respectively.

Proof: For the same reason as discussed in the proof of Lemma 6.3.1, we only need to consider the case where $\{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon,\ x \in \{X_1^k, \ldots, X_{N_k}^k\}\} \neq \emptyset$. Define

$$J_k(\theta) = \frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\big(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}\big) \ln f(X_i^k, \theta), \text{ where } \widetilde{S}_k(\bar{H}_k(x)) := \frac{[S(\bar{H}_k(x))]^k}{\widetilde{f}(x,\theta_k)}.$$

Since $f(\cdot,\theta)$ belongs to the NEF, we can write

$$\begin{aligned} J_k(\theta) &= \frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\big(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}\big) \ln \ell(X_i^k) \\ &\quad + \frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\big(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}\big)\theta^T\Gamma(X_i^k) \\ &\quad - \frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\big(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}\big) \ln \int_{x\in\mathcal{X}} e^{\theta^T\Gamma(x)}\ell(x)\nu(dx). \end{aligned}$$

Thus the gradient of $J_k(\theta)$ with respect to θ can be expressed as

$$\begin{aligned} \nabla_\theta J_k(\theta) &= \frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\big(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}\big)\Gamma(X_i^k) \\ &\quad - \frac{\int e^{\theta^T\Gamma(x)}\Gamma(x)\ell(x)\nu(dx)}{\int e^{\theta^T\Gamma(x)}\ell(x)\nu(dx)}\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\big(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}\big), \end{aligned}$$

where the validity of the interchange of derivative and integral above is guaranteed by Assumption $B6$ and the dominated convergence theorem. By setting $\nabla_\theta J_k(\theta) = 0$, it follows that

$$\frac{\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\left(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}\right)\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}\left(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}\right)} = \frac{\int e^{\theta^T\Gamma(x)}\Gamma(x)\ell(x)\nu(dx)}{\int e^{\theta^T\Gamma(x)}\ell(x)\nu(dx)},$$

which implies that $E_{\widetilde{g}_{k+1}}[\Gamma(X)] = E_\theta[\Gamma(X)]$ by the definitions of $g_k(\cdot)$ (cf. (6.4)) and $f(\cdot,\theta)$.

Since θ_{k+1} is the optimal solution of the problem

$$\arg\max_{\theta\in\Theta} J_k(\theta),$$

we conclude that $E_{\widetilde{g}_{k+1}}\left[\Gamma(X)\right] = E_{\theta_{k+1}}\left[\Gamma(X)\right],\ \forall\ k = 0, 1, \ldots$, by $B5$. ∎

Remark 6.4.3 *Intuitively, the sequence of regions $\{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon\},\ k = 0, 1, 2\ldots$ tends to get smaller and smaller during the search process of SMRAS. Lemma 6.4.2 shows that the sequence of sampling p.d.f's $f(\cdot, \theta_{k+1})$ is "adapted" to this sequence of shrinking regions. For example, consider the special case where $\{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon\}$ is convex and $\Gamma(x) = x$. Since $E_{\widetilde{g}_{k+1}}[X]$ is a convex combination of $X_1^k, \ldots, X_{N_k}^k$, the lemma implies that $E_{\theta_{k+1}}[X] \in \{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon\}$. Thus, it is natural to expect that the random samples generated at the next iteration will fall in the region $\{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon\}$ with large probabilities (e.g., consider the normal distribution where its mean $\mu_{k+1} = E_{\theta_{k+1}}[X]$ is equal to its mode value). In contrast, if we use a fixed sampling distribution for all iterations (cf. e.g., [74], [89]), then sampling from this sequence of shrinking regions could be a substantially difficult problem in practice.*

We now define a sequence of (idealized) p.d.f's $\{g_k(\cdot)\}$ as

$$g_{k+1}(x) = \frac{[S(h(x)]^k\, \widetilde{I}(h(x), \gamma_k)}{\int_{x\in\mathcal{X}} [S(h(x)]^k\, \widetilde{I}(h(x), \gamma_k)\nu(dx)} \quad \forall\ k = 0, 1, \ldots, \tag{6.5}$$

where $\gamma_k := h(X_k^\dagger)$. Notice that since $X_k^\dagger$ is a random variable, $g_{k+1}(x)$ is also random.

The outline of the convergence proof is as follows: First we establish the convergence of the sequence of p.d.f's $\{g_k(\cdot)\}$, then we claim that the reference p.d.f's $\{\widetilde{g}_k(\cdot)\}$ are in fact the (sample average) approximations of the sequence $\{g_k(\cdot)\}$ by showing that $E_{\widetilde{g}_k}[\Gamma(X)] \to E_{g_k}[\Gamma(X)]$ w.p.1 as $k \to \infty$. Thus, the convergence of the sequence $\{f(\cdot, \theta_k)\}$ follows immediately by applying Lemma 6.4.2.

The convergence of the sequence $\{g_k(\cdot)\}$ is formalized in the following lemma.

Lemma 6.4.3 *If Assumptions $L1-L3$, $B1-B3$ are satisfied, then*

$$\lim_{k\to\infty} E_{g_k}\left[\Gamma(X)\right] = \Gamma(x^*) \quad w.p.1.$$

Proof: Our proof is an extension of the proof of Theorem 5.3.1. Let Ω_1 be the set of all sample paths such that step $3a/3b$ is visited finitely often, and let Ω_2 be the set of sample paths such that $\lim_{k\to\infty}\{h(x) \geq \gamma_k - \varepsilon\} \subseteq \Pi$. By Lemma 6.4.1, we have $P(\Omega_1) = 1$, and for each $\omega \in \Omega_1$, there exists a finite $\mathcal{N}(\omega) > 0$ such that

$$X^{\dagger}_{k+1}(\omega) = X^{\dagger}_{k}(\omega) \ \ \forall\, k \geq \mathcal{N}(\omega),$$

which implies that $\gamma_{k+1}(\omega) = \gamma_k(\omega)\ \forall\, k \geq \mathcal{N}(\omega)$. Furthermore, by $B1$, we have $P(\Omega_2) = 1$ and $\{h(x) \geq \gamma_k(\omega) - \varepsilon\} \subseteq \Pi,\ \forall\, k \geq \mathcal{N}(\omega)\ \forall\, \omega \in \Omega_1 \cap \Omega_2$.

Thus, for each $\omega \in \Omega_1 \cap \Omega_2$, it is not difficult to see from equation (6.5) that $g_{k+1}(\cdot)$ can be expressed recursively as

$$g_{k+1}(x) = \frac{S(h(x))g_k(x)}{E_{g_k}[S(h(X))]}, \ \ \forall\ k > \mathcal{N}(\omega),$$

where we have used $g_k(\cdot)$ instead of $g_k(\omega)(\cdot)$ to simplify the notation. It follows that

$$E_{g_{k+1}}\left[S(h(X))\right] = \frac{E_{g_k}[S^2(h(X))]}{E_{g_k}[S(h(X))]} \geq E_{g_k}\left[S(h(X))\right], \ \ \forall\ k > \mathcal{N}(\omega), \tag{6.6}$$

which implies that the sequence $\{E_{g_k}[h(X)], k = 1, 2, \ldots\}$ converges (note that $E_{g_k}[h(X)]$ is bounded from above by $h(x^*)$).

Now we show that the limit of the above sequence is $S(h(x^*))$. To show this, we proceed by contradiction and assume that

$$\lim_{k\to\infty} E_{g_k}\left[S(h(X))\right] = S_* < S^* := S(h(x^*)).$$

Define the set $\mathcal{C} := \{x : h(x) \geq \gamma_{\mathcal{N}(\omega)} - \varepsilon\} \cap \{x : S(h(x)) \geq \frac{S^*+S_*}{2}\} \cap \mathcal{X}$. Since $S(\cdot)$ is strictly increasing, its inverse $S^{-1}(\cdot)$ exists, thus $\mathcal{C}$ can be formulated as $\mathcal{C} = \big\{x : h(x) \geq \max\{\gamma_{\mathcal{N}(\omega)} - \varepsilon, S^{-1}(\frac{S^*+S_*}{2})\}\big\} \cap \mathcal{X}$. By $B2$, $\mathcal{C}$ has a strictly positive Lebesgue/discrete measure.

Note that $g_{k+1}(\cdot)$ can be written as

$$g_{k+1}(x) = \prod_{i=\mathcal{N}(\omega)+1}^{k} \frac{S(h(x))}{E_{g_i}\left[S(h(X))\right]} \cdot g_{\mathcal{N}(\omega)+1}(x), \quad \forall\ k > \mathcal{N}(\omega).$$

Since $\lim_{k\to\infty} \frac{S(h(x))}{E_{g_k}[S(h(X))]} = \frac{S(h(x))}{S_*} > 1,\ \forall\ x \in \mathcal{C}$, we conclude that

$$\liminf_{k\to\infty} g_k(x) = \infty, \quad \forall\ x \in \mathcal{C}.$$

We have, by Fatou's lemma,

$$1 = \liminf_{k\to\infty} \int_{\mathcal{X}} g_{k+1}(x)\nu(dx) \geq \liminf_{k\to\infty} \int_{\mathcal{C}} g_{k+1}(x)\nu(dx) \geq \int_{\mathcal{C}} \liminf_{k\to\infty} g_{k+1}(x)\nu(dx) = \infty,$$

which is a contradiction. Hence, it follows that

$$\lim_{k\to\infty} E_{g_k}\left[S(h(X))\right] = S^*,\ \forall\ \omega \in \Omega_1 \cap \Omega_2. \tag{6.7}$$

We now bound the difference between $E_{g_{k+1}}[\Gamma(X)]$ and $\Gamma(x^*)$. We have

$$\begin{aligned} \|E_{g_{k+1}}\left[\Gamma(X)\right] - \Gamma(x^*)\| &\leq \int_{x\in\mathcal{X}} \|\Gamma(x) - \Gamma(x^*)\| g_{k+1}(x)\nu(dx) \\ &= \int_{\mathcal{D}} \|\Gamma(x) - \Gamma(x^*)\| g_{k+1}(x)\nu(dx), \end{aligned} \tag{6.8}$$

where $\mathcal{D} := \left\{x : h(x) \geq \gamma_{\mathcal{N}(\omega)} - \varepsilon\right\} \cap \mathcal{X}$ is the support of $g_{k+1}(x),\ \forall\ k > \mathcal{N}(\omega)$.

By the assumption on $\Gamma(\cdot)$ in Definition 5.3.1, for any given $\zeta > 0$, there exists a $\delta > 0$ such that $\|x - x^*\| \leq \delta$ implies $\|\Gamma(x) - \Gamma(x^*)\| \leq \zeta$. Let A_δ be defined as in $B3$; then we have from (6.8)

$$\begin{aligned} &\|E_{g_{k+1}}\left[\Gamma(X)\right] - \Gamma(x^*)\| \\ \leq\ & \int_{A_\delta^c \cap \mathcal{D}} \|\Gamma(x) - \Gamma(x^*)\| g_{k+1}(x)\nu(dx) + \int_{A_\delta \cap \mathcal{D}} \|\Gamma(x) - \Gamma(x^*)\| g_{k+1}(x)\nu(dx) \\ \leq\ & \zeta + \int_{A_\delta \cap \mathcal{D}} \|\Gamma(x) - \Gamma(x^*)\| g_{k+1}(x)\nu(dx), \quad \forall\ k > \mathcal{N}(\omega). \end{aligned} \tag{6.9}$$

The rest of the proof amounts to showing that the second term in (6.9) is also bounded. Clearly by $B1$, the term $\|\Gamma(x) - \Gamma(x^*)\|$ is bounded on the set $A_\delta \cap \mathcal{D}$. We only need to find a bound for $g_{k+1}(x)$.

By $B3$, we have

$$\sup_{x \in A_\delta \cap \mathcal{D}} h(x) \leq \sup_{x \in A_\delta} h(x) < h(x^*).$$

Define $S_\delta := S^* - S(\sup_{x \in A_\delta} h(x))$. And by the monotonicity of $S(\cdot)$, we have $S_\delta > 0$. It is easy to see that

$$S(h(x)) \leq S^* - S_\delta, \quad \forall\, x \in A_\delta \cap \mathcal{D}. \tag{6.10}$$

From (6.6) and (6.7), there exists $\bar{\mathcal{N}}(\omega) \geq \mathcal{N}(\omega)$ such that for all $k \geq \bar{\mathcal{N}}(\omega)$

$$E_{g_{k+1}}\left[S(h(X))\right] \geq S^* - \frac{1}{2} S_\delta. \tag{6.11}$$

Observe that $g_{k+1}(x)$ can be rewritten as

$$g_{k+1}(x) = \prod_{i=\bar{\mathcal{N}}}^{k} \frac{S(h(x))}{E_{g_i}\left[S(h(X))\right]} \cdot g_{\bar{\mathcal{N}}}(x), \quad \forall\, k \geq \bar{\mathcal{N}}(\omega).$$

Thus, it follows from (6.10) and (6.11) that

$$g_{k+1}(x) \leq \Big(\frac{S^* - S_\delta}{S^* - \frac{1}{2} S_\delta}\Big)^{k - \bar{\mathcal{N}} + 1} \cdot g_{\bar{\mathcal{N}}}(x), \quad \forall\, x \in A_\delta \cap \mathcal{D}, \quad \forall\, k \geq \bar{\mathcal{N}}(\omega).$$

Therefore,

$$\begin{aligned}
\left\| E_{g_{k+1}}\left[\Gamma(X)\right] - \Gamma(x^*) \right\| &\leq \zeta + \sup_{x \in A_\delta \cap \mathcal{D}} \|\Gamma(x) - \Gamma(x^*)\| \int_{A_\delta \cap \mathcal{D}} g_{k+1}(x) \nu(dx) \\
&\leq \zeta + \sup_{x \in A_\delta \cap \mathcal{D}} \|\Gamma(x) - \Gamma(x^*)\| \Big(\frac{S^* - S_\delta}{S^* - \frac{1}{2} S_\delta}\Big)^{k - \bar{\mathcal{N}} + 1}, \quad \forall\, k \geq \bar{\mathcal{N}}(\omega) \\
&\leq \Big(1 + \sup_{x \in A_\delta \cap \mathcal{D}} \|\Gamma(x) - \Gamma(x^*)\|\Big) \zeta, \quad \forall\, k \geq \widehat{\mathcal{N}}(\omega),
\end{aligned}$$

where $\widehat{\mathcal{N}}(\omega)$ is given by $\widehat{\mathcal{N}}(\omega) := \max\left\{\bar{\mathcal{N}}(\omega),\ \lceil \bar{\mathcal{N}}(\omega) - 1 + \ln \zeta / \ln\left(\frac{S^* - S_\delta}{S^* - \frac{1}{2} S_\delta}\right) \rceil\right\}$.

Since ζ is arbitrary, we have

$$\lim_{k \to \infty} E_{g_k}\left[\Gamma(X)\right] = \Gamma(x^*),\ \forall\, \omega \in \Omega_1 \cap \Omega_2.$$

And since $P(\Omega_1 \cap \Omega_2) = 1$, the proof is thus completed. ∎

As mentioned earlier, the rest of the convergence proof now amounts to showing that $E_{\widetilde{g}_k}[\Gamma(X)] \to E_{g_k}[\Gamma(X)]$ w.p.1 as $k \to \infty$. However, there is one more complication: Since $S(\cdot)$ is an increasing function and is raised to the kth power in both $\widetilde{g}_{k+1}$ and g_{k+1} (cf. (6.4), (6.5)), the associated estimation error between $\bar{H}_k(x)$ and $h(x)$ is exaggerated. Thus, even though we have $\lim_{k\to\infty} \bar{H}_k(x) = h(x)$ w.p.1, the quantities $S^k(\bar{H}_k(x))$ and $S^k(h(x))$ may still differ considerably as k gets large. Therefore, the sequence $\{\bar{H}_k(x)\}$ not only has to converge to $h(x)$, but it should also do so at a fast enough rate in order to reduce the gap between $S^k(\bar{H}_k(x))$ and $S^k(h(x))$. This requirement is summarized in the following assumption.

Assumption *L*4. *For any given $\zeta > 0$, there exist $\delta^* \in (0,1)$ and $\mathcal{K} > 0$ such that the observation allocation rule $\{M_k, k = 1, 2\ldots\}$ satisfies*

$$\alpha^k \phi\Big(M_k, \min\big\{\Delta_k, \frac{\zeta}{\alpha^k}, \frac{\zeta}{\alpha^k L_k}\big\}\Big) \le (\delta^*)^k \;\forall\; k \ge \mathcal{K},$$

where $\phi(\cdot,\cdot)$ is defined as in $L1$, Δ_k and L_k are defined as in $B4$.

Let $S(z) = e^{\tau z}$, for some positive constant τ. We have $S^k(z) = e^{\tau k z}$ and $[S^k(z)]' = k\tau e^{\tau k z}$. It is easy to verify that $\frac{|S^k(z)-S^k(\bar{z})|}{S^k(z)} \le k\tau e^{\tau k \Delta_k}|z - \bar{z}| \;\forall\, \bar{z} \in (z - \Delta_k, z + \Delta_k)$, and $B4$ is satisfied for $\Delta_k = 1/k$ and $L_k = \tau e^{\tau} k$. Thus, the condition in $L4$ becomes $\alpha^k \phi(M_k, \bar{\zeta}/\alpha^k k) \le (\delta^*)^k \;\forall\; k \ge \mathcal{K}$, where $\bar{\zeta} = \zeta/\tau e^{\tau}$. We consider the following two special cases of $L4$. Let $H_i(x)$ be i.i.d. with $E(H_i(x)) = h(x)$ and uniformly bounded variance $\sup_{x\in\mathcal{X}} \sigma^2(x) \le \sigma^2$. By Chebyshev's inequality

$$P\Big(\big|\bar{H}_k(x) - h(x)\big| \ge \frac{\bar{\zeta}}{\alpha^k k}\Big) \le \frac{\sigma^2 \alpha^{2k} k^2}{M_k \bar{\zeta}^2}.$$

Thus, it is easy to check that $L4$ is satisfied by $M_k = (\mu\alpha^3)^k$ for any constant $\mu > 1$.

As a second example, consider the case where $H_1(x), \ldots, H_{N_k}(x)$ are i.i.d. with

$E(H_i(x)) = h(x)$ and bounded support $[a, b]$. By the Hoeffding inequality ([40])

$$P\Big(\big|\bar{H}_k(x) - h(x)\big| \geq \frac{\bar{\zeta}}{\alpha^k k}\Big) \leq 2\exp\Big(\frac{-2M_k\bar{\zeta}^2}{(b-a)^2\alpha^{2k}k^2}\Big).$$

In this case, $L4$ is satisfied by $M_k = (\mu\alpha^2)^k$ for any constant $\mu > 1$.

Again, as discussed in Remark 6.4.2, Assumption $L4$ can be replaced by the weaker condition

$$\sum_{k=1}^{\infty} \phi\Big(M_k, \min\big\{\Delta_k, \frac{\zeta}{\alpha^k}, \frac{\zeta}{\alpha^k L_k}\big\}\Big) < \infty$$

when the solution space $\mathcal{X}$ is discrete finite.

Proposition 6.4.1 *If Assumptions $L1-L4$ are satisfied, then*

$$\lim_{k\to\infty} \alpha^k \big|\bar{\gamma}_{k+1} - \gamma_k\big| = 0 \quad w.p.1.$$

Proof: Again, we consider the sequence $\{X_k^\dagger\}$ generated by SMRAS.

We have for any $\zeta > 0$

$$\begin{aligned}
P\Big(\big|\bar{\gamma}_{k+1} - \gamma_{k+1}\big| \geq \frac{\zeta}{\alpha^k}\Big) &= P\Big(\big|\bar{H}_k(X_{k+1}^\dagger) - h(X_{k+1}^\dagger)\big| \geq \frac{\zeta}{\alpha^k}\Big) \\
&\leq P\Big(\bigcup_{x\in\Lambda_k} \big\{\big|\bar{H}_k(x) - h(x)\big| \geq \frac{\zeta}{\alpha^k}\big\}\Big) \\
&\leq \sum_{x\in\Lambda_k} P\Big(\big|\bar{H}_k(x) - h(x)\big| \geq \frac{\zeta}{\alpha^k}\Big) \\
&\leq \big|\Lambda_k\big| \sup_{x\in\mathcal{X}} P\Big(\big|\bar{H}_k(x) - h(x)\big| \geq \frac{\zeta}{\alpha^k}\Big) \\
&\leq \alpha^k N_0 \phi(M_k, \zeta/\alpha^k) \ \text{ by } L1 \\
&\leq N_0(\delta^*)^k \ \forall\, k \geq \mathcal{K} \ \text{ by } L4 \text{ and the definition of } \phi(\cdot,\cdot).
\end{aligned}$$

Thus

$$\sum_{k=1}^{\infty} P\Big(\big|\bar{\gamma}_{k+1} - \gamma_{k+1}\big| \geq \frac{\zeta}{\alpha^k}\Big) \leq \mathcal{K} + N_0 \sum_{k=\mathcal{K}}^{\infty} (\delta^*)^k < \infty.$$

And by Borel-Cantelli lemma,

$$P\Big(\big\{\big|\bar{\gamma}_{k+1}-\gamma_{k+1})\big|\geq \frac{\zeta}{\alpha^k}\big\}\ i.o.\Big)=0.$$

Let Ω_1 be defined as before, and define $\Omega_3 := \big\{\omega : \alpha^k\,|\bar{\gamma}_{k+1}-\gamma_{k+1}|\geq \zeta\ i.o.\big\}$. Since for each $\omega\in\Omega_1$, there exists a finite $\mathcal{N}(\omega)>0$ such that $\gamma_{k+1}(\omega)=\gamma_k(\omega)\ \forall\, k\geq \mathcal{N}(\omega)$, we have

$$\begin{aligned}
& P\Big(\alpha^k\big|\bar{\gamma}_{k+1}-\gamma_k\big|\geq\zeta\ i.o.\Big)\\
= \;& P\Big(\big|\bar{\gamma}_{k+1}-\gamma_k\big|\geq\frac{\zeta}{\alpha^k}\ i.o.\cap\Omega_1\Big)+P\Big(\big|\bar{\gamma}_{k+1}-\gamma_k\big|\geq\frac{\zeta}{\alpha^k}\ i.o.\cap\Omega_1^c\Big)\\
\leq \;& P(\Omega_3\cap\Omega_1)+P(\Omega_1^c)\\
= \;& 0.
\end{aligned}$$

And since ζ is arbitrary, the proof is thus completed. ∎

We are now ready to state the main theorem.

Theorem 6.4.1 *Let $\varphi>0$ be a positive constant satisfying the condition that the set $\big\{x : S(h(x))\geq\frac{1}{\varphi}\big\}$ has a strictly positive Lebesgue/counting measure. If assumptions L1–L4, B1–B7 are satisfied, and there exist $\delta\in(0,1)$ and $\mathcal{T}_\delta<\infty$ such that $\alpha\geq[\varphi S^*]^2/[\lambda_k^{2/k}\delta]\ \forall\, k\geq\mathcal{T}_\delta$, then*

$$\lim_{k\to\infty}E_{\theta_k}\left[\Gamma(X)\right]=\Gamma(x^*)\quad w.p.1, \tag{6.12}$$

where the limit above is component-wise.

Remark 6.4.4 *By the monotonicity of $S(\cdot)$ and Assumption B2, it is easy to see that such a positive constant φ in Theorem 6.4.1 always exists. Moreover, for continuous problems, φ can be chosen such that $\varphi S^*\approx 1$; for discrete problems, if the counting measure is used, then we can choose $\varphi=1/S^*$.*

Remark 6.4.5 *Note that when $\Gamma(x)$ is a one-to-one function, the above result can be equivalently written as $\Gamma^{-1}\left(\lim_{k\to\infty} E_{\theta_k}\left[\Gamma(X)\right]\right) = x^*$. Also note that for some particular p.d.f.'s/p.m.f.'s, the solution vector x itself will be a component of $\Gamma(x)$ (e.g., multivariate normal p.d.f.). Under these circumstances, we can disregard the redundant components and interpret (6.12) as $\lim_{k\to\infty} E_{\theta_k}\left[X\right] = x^*$. Another special case of particular interest is when the components of the random vector $X = (X_1, \dots, X_n)$ are independent, and each has a univariate p.d.f./p.m.f. of the form*

$$f(x_i, \vartheta_i) = \exp(x_i\vartheta_i - K(\vartheta_i))\ell(x_i),\ \vartheta_i \subset \Re, \forall\ i = 1, \dots, n.$$

In this case, since the distribution of the random vector X is simply the product of the marginal distributions, we have $\Gamma(x) = x$. Thus, equation (6.12) is again equivalent to $\lim_{k\to\infty} E_{\theta_k}\left[X\right] = x^*$, where $\theta_k := (\vartheta_1^k, \dots, \vartheta_n^k)$, and ϑ_i^k is the value of ϑ_i at the kth iteration of the algorithm. The above observations indicate that the convergence result in Theorem 6.4.1 is much stronger than it appears to be.

Proof: For brevity, we define the function

$$Y_k(Z,\gamma) := \widetilde{S}_k(Z)\widetilde{I}(Z,\gamma), \quad \text{where } \widetilde{S}_k(Z) = \begin{cases} [S(h(x))]^k/\widetilde{f}(x,\theta_k) & \text{if } Z = h(x), \\ [S(\bar{H}_k(x))]^k/\widetilde{f}(x,\theta_k) & \text{if } Z = \bar{H}_k(x). \end{cases}$$

By $B7$, the support of $\widetilde{f}(\cdot,\theta_k)$ satisfies $\mathcal{X} \subseteq supp\{\widetilde{f}(\cdot,\theta_k)\}\ \forall\, k$. Thus, we can write

$$E_{g_{k+1}}[\Gamma(X)] = \frac{\widetilde{E}_{\theta_k}[Y_k(h(X),\gamma_k)\Gamma(X)]}{\widetilde{E}_{\theta_k}[Y_k(h(X),\gamma_k)]},$$

where $\widetilde{E}_{\theta_k}(\cdot)$ is the expectation taken with respect to $\widetilde{f}(\cdot,\theta_k)$. We now show $E_{\widetilde{g}_{k+1}}[\Gamma(X)] \to E_{g_{k+1}}[\Gamma(X)]$ w.p.1 as $k \to \infty$. Since we are only interested in the limiting behavior of $E_{\widetilde{g}_{k+1}}[\Gamma(X)]$, from the definition of $\widetilde{g}_{k+1}(\cdot)$ (cf. (6.4)), it is sufficient to show that

$$\frac{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})\Gamma(X_i^k)}{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})} \to E_{g_{k+1}}[\Gamma(X)] \quad w.p.1,$$

where and hereafter, whenever $\Lambda_k \cap \{x : \bar{H}_k(x) > \bar{\gamma}_{k+1} - \varepsilon\} = \emptyset$, we define $0/0 = 0$. We have

$$\begin{aligned}
&\frac{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})\Gamma(X_i^k)}{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})} - E_{g_{k+1}}[\Gamma(X)] \\
= \ & \frac{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})\Gamma(X_i^k)}{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})} - \frac{\widetilde{E}_{\theta_k}[Y_k(h(X), \gamma_k)\Gamma(X)]}{\widetilde{E}_{\theta_k}[Y_k(h(X), \gamma_k)]} \\
= \ & \left\{ \frac{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})\Gamma(X_i^k)}{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})} - \frac{\sum_{i=1}^{N_k} Y_k(h(X_i^k), \gamma_k)\Gamma(X_i^k)}{\sum_{i=1}^{N_k} Y_k(h(X_i^k), \gamma_k)} \right\} \\
& + \left\{ \frac{\frac{1}{N_k}\sum_{i=1}^{N_k} Y_k(h(X_i^k), \gamma_k)\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k} Y_k(h(X_i^k), \gamma_k)} - \frac{\widetilde{E}_{\theta_k}[Y_k(h(X), \gamma_k)\Gamma(X)]}{\widetilde{E}_{\theta_k}[Y_k(h(X), \gamma_k)]} \right\} \\
= \ & \left\{ \frac{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})\Gamma(X_i^k)}{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})} - \frac{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \gamma_k)\Gamma(X_i^k)}{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \gamma_k)} \right\} \quad [i] \\
& + \left\{ \frac{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \gamma_k)\Gamma(X_i^k)}{\sum_{i=1}^{N_k} Y_k(\bar{H}_k(X_i^k), \gamma_k)} - \frac{\sum_{i=1}^{N_k} Y_k(h(X_i^k), \gamma_k)\Gamma(X_i^k)}{\sum_{i=1}^{N_k} Y_k(h(X_i^k), \gamma_k)} \right\} \quad [ii] \\
& + \left\{ \frac{\frac{1}{N_k}\sum_{i=1}^{N_k} Y_k(h(X_i^k), \gamma_k)\Gamma(X_i^k)}{\frac{1}{N_k}\sum_{i=1}^{N_k} Y_k(h(X_i^k), \gamma_k)} - \frac{\widetilde{E}_{\theta_k}[Y_k(h(X), \gamma_k)\Gamma(X)]}{\widetilde{E}_{\theta_k}[Y_k(h(X), \gamma_k)]} \right\} \quad [iii].
\end{aligned}$$

We now analyze the terms $[i] - [iii]$.

(1). We define $\mathcal{E}_k := \{x : \bar{H}_k(x) > \min(\bar{\gamma}_{k+1}, \gamma_k) - \varepsilon, \ x \in \Lambda_k\}$. Note that if $\mathcal{E}_k = \emptyset$, then $[i] = 0$ by convention. When $\mathcal{E}_k \neq \emptyset$, we let $\bar{\eta}_k := 1/\max_{x \in \mathcal{E}_k} \widetilde{S}_k(\bar{H}_k(x))$. Thus

$$[i] = \frac{\sum_{i=1}^{N_k} \bar{\eta}_k \widetilde{S}_k(\bar{H}(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})\Gamma(X_i^k)}{\sum_{i=1}^{N_k} \bar{\eta}_k \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})} - \frac{\sum_{i=1}^{N_k} \bar{\eta}_k \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \gamma_k)\Gamma(X_i^k)}{\sum_{i=1}^{N_k} \bar{\eta}_k \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \gamma_k)}.$$

We have

$$\begin{aligned}
& \Big| \sum_{i=1}^{N_k} \bar{\eta}_k \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}) - \sum_{i=1}^{N_k} \bar{\eta}_k \widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \gamma_k) \Big| \\
\leq \ & \sum_{i=1}^{N_k} \Big| \widetilde{I}(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1}) - \widetilde{I}(\bar{H}_k(X_i^k), \gamma_k) \Big| \quad \text{since } \bar{\eta}_k \widetilde{S}_k(\bar{H}_k(x)) \leq 1 \ \forall x \in \mathcal{E}_k \\
\leq \ & \alpha^k N_0 \frac{1}{\varepsilon} |\bar{\gamma}_{k+1} - \gamma_k| \quad \text{by the definition of } \widetilde{I}(\cdot, \cdot) \\
\longrightarrow \ & 0 \quad w.p.1 \quad \text{by Proposition 6.4.1.}
\end{aligned}$$

Similar argument can also be used to show that w.p.1

$$\Big|\sum_{i=1}^{N_k}\bar{\eta}_k\widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\bar{\gamma}_{k+1})\Gamma(X_i^k)-\sum_{i=1}^{N_k}\bar{\eta}_k\widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)\Gamma(X_i^k)\Big|\to 0.$$

Therefore, $[i]\to 0$ as $k\to\infty$ w.p.1.

(2). Define $\bar{\mathcal{E}}_k := \{x : h(x) > \gamma_k-\varepsilon,\ x\in\Lambda_k\}\cup\{x : \bar{H}_k(x) > \gamma_k-\varepsilon,\ x\in\Lambda_k\}$. If $\bar{\mathcal{E}}_k=\emptyset$, then $[ii]=0$ by convention. If $\bar{\mathcal{E}}_k\neq\emptyset$, we let $\eta_k := 1/\max_{x\in\bar{\mathcal{E}}_k}\widetilde{S}_k(h(x))$, thus

$$[ii]=\frac{\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)\Gamma(X_i^k)}{\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)}-\frac{\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(h(X_i^k))\widetilde{I}(h(X_i^k),\gamma_k)\Gamma(X_i^k)}{\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(h(X_i^k))\widetilde{I}(h(X_i^k),\gamma_k)}.$$

And it is not difficult to see that we will have either $\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)\geq 1$ or $\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(h(X_i^k))\widetilde{I}(h(X_i^k),\gamma_k)\geq 1$ or both. Therefore, in order to prove that $[ii]\to 0$ w.p.1, it is sufficient to show that w.p.1

$$\big|\textstyle\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)-\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(h(X_i^k))\widetilde{I}(h(X_i^k),\gamma_k)\big|\to 0\ \text{ and}$$

$$\big|\textstyle\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)\Gamma(X_i^k)-\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(h(X_i^k))\widetilde{I}(h(X_i^k),\gamma_k)\Gamma(X_i^k)\big|\to 0.$$

We have

$$\begin{aligned}&\Big|\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)-\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(h(X_i^k))\widetilde{I}(h(X_i^k),\gamma_k)\Big|\\ \leq\ &\Big|\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)-\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(h(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)\Big|\quad [a]\\ &+\Big|\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(h(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)-\sum_{i=1}^{N_k}\eta_k\widetilde{S}_k(h(X_i^k))\widetilde{I}(h(X_i^k),\gamma_k)\Big|\quad [b]\end{aligned}$$

$$\begin{aligned}[a]\ &\leq\ \sum_{i=1}^{N_k}\frac{\big|\widetilde{S}_k(\bar{H}_k(X_i^k))-\widetilde{S}_k(h(X_i^k))\big|}{\widetilde{S}_k(h(X_i^k))}\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k)\\ &=\ \sum_{i=1}^{N_k}\frac{\big|[S(\bar{H}_k(X_i^k))]^k-[S(h(X_i^k))]^k\big|}{[S(h(X_i^k))]^k}\widetilde{I}(\bar{H}_k(X_i^k),\gamma_k).\end{aligned}\qquad(6.13)$$

Note that

$$\begin{aligned}
P\left(\max_{1\le i\le N_k}\left|\bar{H}_k(X_i^k)-h(X_i^k)\right|\ge\Delta_k\right) &\le P\Big(\bigcup_{x\in\Lambda_k}\left\{\left|\bar{H}_k(x)-h(x)\right|\ge\Delta_k\right\}\Big),\\
&\le \sum_{x\in\Lambda_k}P\left(\left|\bar{H}_k(x)-h(x)\right|\ge\Delta_k\right),\\
&\le |\Lambda_k|\sup_{x\in\mathcal{X}}P\left(\left|\bar{H}_k(x)-h(x)\right|\ge\Delta_k\right),\\
&\le \alpha^k N_0\phi(M_k,\Delta_k)\ \text{ by } L1,\\
&\le N_0(\delta^*)^k\ \ \forall\ k\ge\mathcal{K}\ \text{ by } L4.
\end{aligned}$$

Furthermore,

$$\sum_{k=1}^{\infty}P\left(\max_{1\le i\le N_k}\left|\bar{H}_k(X_i^k)-h(X_i^k)\right|\ge\Delta_k\right)\le\mathcal{K}+N_0\sum_{k=\mathcal{K}}^{\infty}(\delta^*)^k<\infty,$$

which implies that $P\left(\max_{1\le i\le N_k}\left|\bar{H}_k(X_i^k)-h(X_i^k)\right|\ge\Delta_k\ i.o.\right)=0$ by the Borel-Cantelli lemma.

Let $\Omega_4:=\{\omega:\max_{1\le i\le N_k}\left|\bar{H}_k(X_i^k)-h(X_i^k)\right|<\Delta_k\ i.o.\}$. For each $\omega\in\Omega_4$, we have

$$\begin{aligned}
(6.13)\ &\le \sum_{i=1}^{N_k}L_k\left|\bar{H}_k(X_i^k)-h(X_i^k)\right|\ \text{ for sufficiently large } k\text{, by } B4,\\
&\le \alpha^k N_0 L_k\max_{1\le i\le N_k}\left|\bar{H}_k(X_i^k)-h(X_i^k)\right|\ \text{ for sufficiently large } k.
\end{aligned}$$

Notice that for any given $\zeta>0$,

$$P\Big\{\alpha^k L_k\max_{1\le i\le N_k}\left|\bar{H}_k(X_i^k)-h(X_i^k)\right|\ge\zeta\Big\}\le P\Big(\bigcup_{x\in\Lambda_k}\{\left|\bar{H}_k(x)-h(x)\right|\ge\frac{\zeta}{\alpha^k L_k}\}\Big).$$

And by using $L4$ and a similar argument as in the proof for Proposition 6.4.1, it is easy to show that

$$\alpha^k L_k\max_{1\le i\le N_k}\left|\bar{H}_k(X_i^k)-h(X_i^k)\right|\to 0\ \ w.p.1$$

Let $\Omega_5:=\left\{\omega:\alpha^k L_k\max_{1\le i\le N_k}\left|\bar{H}_k(X_i^k)-h(X_i^k)\right|\to 0\right\}$. Since $P(\Omega_4\cap\Omega_5)\ge 1-P(\Omega_4^c)-P(\Omega_5^c)=1$, it follows that $[a]\to 0$ as $k\to\infty$ w.p.1.

On the other hand,

$$\begin{aligned}
[b] \quad &\leq \quad \sum_{i=1}^{N_k} \eta_k \widetilde{S}_k(h(X_i^k)) \left| \widetilde{I}(\bar{H}_k(X_i^k), \gamma_k) - \widetilde{I}(h(X_i^k), \gamma_k) \right| \\
&\leq \quad \alpha^k N_0 \frac{1}{\varepsilon} \max_{1 \leq i \leq N_k} \left| \bar{H}_k(X_i^k) - h(X_i^k) \right| \\
&\longrightarrow \quad 0 \;\; w.p.1 \;\; \text{by a similar argument as before.}
\end{aligned}$$

By repeating the above argument, we can also show that

$$\Big| \sum_{i=1}^{N_k} \eta_k \widetilde{S}_k(\bar{H}_k(X_i^k)) \widetilde{I}(\bar{H}_k(X_i^k), \gamma_k) \Gamma(X_i^k) - \sum_{i=1}^{N_k} \eta_k \widetilde{S}_k(h(X_i^k)) \widetilde{I}(h(X_i^k), \gamma_k) \Gamma(X_i^k) \Big| \to 0 \;\; w.p.1.$$

Hence, we have $[ii] \to 0$ as $k \to \infty$ w.p.1.

(3).

$$[iii] = \frac{\frac{1}{N_k} \sum_{i=1}^{N_k} \varphi^k \widetilde{S}_k(h(X_i^k)) \widetilde{I}(h(X_i^k), \gamma_k) \Gamma(X_i^k)}{\frac{1}{N_k} \sum_{i=1}^{N_k} \varphi^k \widetilde{S}_k(h(X_i^k)) \widetilde{I}(h(X_i^k), \gamma_k)} - \frac{\widetilde{E}_{\theta_k}\left[\varphi^k \widetilde{S}_k(h(X)) \widetilde{I}(h(X), \gamma_k) \Gamma(X)\right]}{\widetilde{E}_{\theta_k}\left[\varphi^k \widetilde{S}_k(h(X)) \widetilde{I}(h(X), \gamma_k)\right]}.$$

Since $\varepsilon > 0$, we have $\gamma_k - \varepsilon \leq h(x^*) - \varepsilon$ for all k. Thus by $B2$, the set $\{x : h(x) \geq \gamma_k - \varepsilon\} \cap \mathcal{X}$ has a strictly positive Lebesgue/discrete measure for all k. It follows from Fatou's lemma that

$$\liminf_{k \to \infty} \widetilde{E}_{\theta_k}\left[\varphi^k \widetilde{S}_k(h(X)) \widetilde{I}(h(X), \gamma_k)\right] \geq \int_{\mathcal{X}} \liminf_{k \to \infty} [\varphi S(h(x))]^k \widetilde{I}(h(x), \gamma_k) \nu(dx) > 0,$$

where the last inequality follows from $\varphi S(h(x)) \geq 1 \; \forall x \in \big\{x : h(x) \geq \max\{S^{-1}(\frac{1}{\varphi}), h(x^*) - \varepsilon\}\big\}$.

We denote by $\mathcal{U}_k$ the event that the total number of visits to step $3a/3b$ is less than or equal to $\sqrt{k}$ at the kth iteration of the algorithm, and by $\mathcal{V}_k$ the event that $\{h(x) \geq \gamma_k - \varepsilon\} \subseteq \Pi$. And for any $\xi > 0$, let $\mathcal{C}_k$ be the event

$$\Big| \frac{1}{N_k} \sum_{i=1}^{N_k} \varphi^k \widetilde{S}_k(h(X_i^k)) \widetilde{I}(h(X_i^k), \gamma_k) - \widetilde{E}_{\theta_k}\left[\varphi^k \widetilde{S}_k(h(X)) \widetilde{I}(h(X), \gamma_k)\right] \Big| \geq \xi.$$

Note that we have $P(\mathcal{U}_k^c \text{ } i.o.) = 0$ by Lemma 6.4.1, and $P(\mathcal{V}_k^c \text{ } i.o.) = 0$ by $B1$. Therefore,

$$\begin{aligned} P\left(\mathcal{C}_k \ i.o.\right) &= P\big(\{\mathcal{C}_k \cap \mathcal{U}_k\} \cup \{\mathcal{C}_k \cap \mathcal{U}_k^c\} \ i.o.\big) \\ &= P\big(\mathcal{C}_k \cap \mathcal{U}_k \ i.o.\big) \\ &= P\big(\{\mathcal{C}_k \cap \mathcal{U}_k \cap \mathcal{V}_k\} \cup \{\mathcal{C}_k \cap \mathcal{U}_k \cap \mathcal{V}_k^c\} \ i.o.\big) \\ &= P\big(\mathcal{C}_k \cap \mathcal{U}_k \cap \mathcal{V}_k \ i.o.\big). \end{aligned} \tag{6.14}$$

From $B7$, it is easy to see that conditional on the event $\mathcal{V}_k$, the support $[a_k, b_k]$ of the random variable $\varphi^k \widetilde{S}_k(h(X_i^k))\widetilde{I}(h(X_i^k), \gamma_k)$ satisfies $[a_k, b_k] \subseteq \big[0, \frac{(\varphi S^*)^k}{\lambda_k f_*}\big]$. Moreover, conditional on θ_k and γ_k, $X_1^k, \ldots, X_{N_k}^k$ are i.i.d. random variables with common density $\widetilde{f}(\cdot, \theta_k)$, we have by the Hoeffding inequality,

$$\begin{aligned} P\big(\mathcal{C}_k \big| \mathcal{V}_k, \theta_k = \theta, \gamma_k = \gamma\big) &\leq 2\exp\Big(\frac{-2N_k\xi^2}{(b_k - a_k)^2}\Big) \\ &\leq 2\exp\Big(\frac{-2N_k\xi^2\lambda_k^2 f_*^2}{(\varphi S^*)^{2k}}\Big) \quad \forall\ k = 1, 2, \ldots. \end{aligned}$$

Thus,

$$\begin{aligned} P\left(\mathcal{C}_k \cap \mathcal{V}_k\right) &= \int_{\theta,\gamma} P\left(\mathcal{C}_k \cap \mathcal{V}_k \big| \theta_k = \theta, \gamma_k = \gamma\right) f_{\theta_k,\gamma_k}(d\theta, d\gamma) \\ &= \int_{\theta,\mathcal{V}_k} P\left(\mathcal{C}_k \big| \mathcal{V}_k, \theta_k = \theta, \gamma_k = \gamma\right) f_{\theta_k,\gamma_k}(d\theta, d\gamma) \\ &\leq 2\exp\Big(\frac{-2N_k\xi^2\lambda_k^2 f_*^2}{(\varphi S^*)^{2k}}\Big), \end{aligned}$$

where $f_{\theta_k,\gamma_k}(\cdot,\cdot)$ is the joint distribution of random variables θ_k and γ_k. It follows that

$$\begin{aligned} P\left(\mathcal{C}_k \cap \mathcal{U}_k \cap \mathcal{V}_k\right) &\leq P\left(\mathcal{C}_k \cap \mathcal{V}_k \big| \mathcal{U}_k\right) \\ &\leq 2\exp\Big(\frac{-2\alpha^{k-\sqrt{k}} N_0\xi^2\lambda_k^2 f_*^2}{(\varphi S^*)^{2k}}\Big) \\ &\leq 2\exp\Big(\frac{-2N_0\xi^2 f_*^2}{\alpha^{\sqrt{k}}}\Big(\frac{\alpha\lambda_k^{2/k}}{(\varphi S^*)^2}\Big)^k\Big), \end{aligned}$$

where the second inequality above follows from the fact that conditional on $\mathcal{U}_k$, the total number of visits to step $3c$ is greater than $k - \sqrt{k}$.

Moreover, since $e^{-x} < 1/x \; \forall \; x > 0$, we have

$$P\left(\mathcal{C}_k \cap \mathcal{U}_k \cap \mathcal{V}_k\right) < \frac{\alpha^{\sqrt{k}}}{N_0 \xi^2 f_*^2} \Big(\frac{(\varphi S^*)^2}{\alpha \lambda_k^{2/k}} \Big)^k = \frac{1}{N_0 \xi^2 f_*^2} \Big(\frac{\alpha^{\sqrt{k}/k} (\varphi S^*)^2}{\alpha \lambda_k^{2/k}} \Big)^k.$$

By assumption, we have $\frac{(\varphi S^*)^2}{\alpha \lambda_k^{2/k}} \leq \delta < 1$ for all $k \geq \mathcal{T}_\delta$. Thus, there exist $\delta < \widetilde{\delta} < 1$ and $\mathcal{T}_{\widetilde{\delta}} > 0$ such that $\alpha^{\sqrt{k}/k} \frac{(\varphi S^*)^2}{\alpha \lambda_k^{2/k}} \leq \widetilde{\delta} \; \forall \, k \geq \mathcal{T}_{\widetilde{\delta}}$. Therefore,

$$\sum_{k=1}^{\infty} P\left(\mathcal{C}_k \cap \mathcal{U}_k \cap \mathcal{V}_k\right) < \mathcal{T}_{\widetilde{\delta}} + \frac{1}{N_0 \xi^2 f_*^2} \sum_{k=\mathcal{T}_{\widetilde{\delta}}}^{\infty} \widetilde{\delta}^k < \infty.$$

Thus, we have by the Borel-Cantelli lemma

$$P\left(\mathcal{C}_k \cap \mathcal{U}_k \cap \mathcal{V}_k \; i.o.\right) = 0,$$

which implies that $P(\mathcal{C}_k \; i.o.) = 0$ by (6.14). And since $\xi > 0$ is arbitrary, we have

$$\Big| \frac{1}{N_k} \sum_{i=1}^{N_k} \varphi^k \widetilde{S}_k(h(X_i^k)) \widetilde{I}(h(X_i^k), \gamma_k) - \widetilde{E}_{\theta_k} \Big[\varphi^k \widetilde{S}_k(h(X)) \widetilde{I}(h(X), \gamma_k) \Big] \Big| \to 0 \;\; w.p.1. \text{ as } k \to \infty.$$

The same argument can also be used to show that

$$\Big| \frac{1}{N_k} \sum_{i=1}^{N_k} \varphi^k \widetilde{S}_k(h(X_i^k)) \widetilde{I}(h(X_i^k), \gamma_k) \Gamma(X_i^k) - \widetilde{E}_{\theta_k} \big[\varphi^k \widetilde{S}_k(h(X)) \widetilde{I}(h(X), \gamma_k) \Gamma(X) \big] \Big| \to 0 \;\; w.p.1.$$

And because $\liminf_{k\to\infty} \widetilde{E}_{\theta_k} \left[\varphi^k \widetilde{S}_k(h(X)) \widetilde{I}(h(X), \gamma_k) \right] > 0$, we have $[iii] \to 0$ w.p.1 as $k \to \infty$.

Hence the proof is completed by applying Lemma 6.4.2 and 6.4.3. ∎

We now address some of the special cases discussed in Remark 6.4.5; the proofs are straightforward and hence omitted.

Corollary 6.4.2 (Multivariate Normal) *For continuous optimization problems in $\Re^n$, if multivariate normal p.d.f.'s are used in SMRAS, i.e.,*

$$f(x, \theta_k) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_k|}} \exp\Big(-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) \Big),$$

where $\theta_k := (\mu_k;\ \Sigma_k)$, assumptions $L1-L4$, $B1-B5$ are satisfied, and there exist $\delta \in (0,1)$ and $\mathcal{T}_\delta < \infty$ such that $\alpha \geq [\varphi S^*]^2/[\lambda_k^{2/k}\delta]\ \forall\, k \geq \mathcal{T}_\delta$, then

$$\lim_{k\to\infty} \mu_k = x^*, \quad \textit{and} \quad \lim_{k\to\infty} \Sigma_k = 0_{n\times n} \quad \textit{w.p.1},$$

where $0_{n\times n}$ represents an n-by-n zero matrix.

Corollary 6.4.3 (Independent Univariate) *If the components of the random vector $X = (X_1, \ldots, X_n)$ are independent, each has a univariate p.d.f./p.m.f. of the form*

$$f(x_i, \vartheta_i) = \exp(x_i\vartheta_i - K(\vartheta_i))\ell(x_i),\ \vartheta_i \subset \Re, \forall\ i = 1, \ldots, n,$$

assumptions $L1 - L4$, $B1 - B7$ are satisfied, and there exist $\delta \in (0,1)$ and $\mathcal{T}_\delta < \infty$ such that $\alpha \geq [\varphi S^*]^2/[\lambda_k^{2/k}\delta]\ \forall\ k \geq \mathcal{T}_\delta$, then

$$\lim_{k\to\infty} E_{\theta_k}[X] = x^* \quad \textit{w.p.1}, \quad \textit{where} \quad \theta_k := (\vartheta_1^k, \ldots, \vartheta_n^k).$$

Remark 6.4.6 (Stopping Rule): *We now return to the issue of designing a valid stopping rule for SMRAS. In practice, this can be achieved in many different ways. The simplest method is to stop the algorithm when the total computational budget is exhausted or when the prescribed maximum number of iterations is reached. Since Proposition 6.4.1 indicates that the sequence $\{\bar{\gamma}_k,\ k = 0, 1, \ldots\}$ generated by SMRAS converges, an alternative stopping criteria could be based on identifying whether the sequence has settled down to its limit value. To do so, we consider the moving average process $\{\Upsilon_k^{(l)}\}$ defined as follows*

$$\Upsilon_k^{(l)} := \frac{1}{l}\sum_{i=k-l+1}^{k} \bar{\gamma}_i, \quad \forall\ k \geq l-1,$$

where $l \geq 1$ is a predefined constant. It is easy to see that an unbiased estimator of the sample variance of $\Upsilon_k^{(l)}$ is

$$\widetilde{\mathbf{var}}(\Upsilon_k^{(l)}) := \frac{\sum_{i=k-l+1}^{k}[\bar{\gamma}_i - \Upsilon_k^{(l)}]^2}{l(l-1)},$$

which approaches zero as the sequence $\{\bar{\gamma}_k\}$ approaches its limit. Thus, a reasonable approach in practice is to stop the algorithm when the value of $\widetilde{\mathbf{var}}(\Upsilon_k^{(l)})$ falls below some pre-specified tolerance level, i.e., $\exists\, k > 0$ such that $\widetilde{\mathbf{var}}(\Upsilon_k^{(l)}) \le \tau$, where $\tau > 0$ is the tolerance level.

6.5 Numerical Examples

In this Chapter, we test the performance of SMRAS on both continuous and combinatorial stochastic optimization problems. In the former case, we first illustrate the global convergence of SMRAS by testing the algorithm on two multi-extremal functions; then we apply the algorithm to an inventory control problem. In the latter case, we consider the problem of optimizing the buffer allocations in a tandem queue with unreliable servers, which has been previously studied in e.g., [3], [84].

We now discuss some implementation issues of SMRAS.

1. Since SMRAS was presented in a maximization context, the following slight modifications are required before it can be applied to minimization problems: (*i*) $S(\cdot)$ needs to be initialized as a strictly decreasing function instead of strictly increasing. Throughout this Chapter, we take $S(z) := \beta^z$ for maximization problems and $S(z) := \beta^{-z}$ for minimization problems, where $\beta > 1$ is some predefined constant. (*ii*) The sample $(1-\rho_k)$-quantile $\widetilde{\gamma}_{k+1}$ will now be calculated by first ordering the sample performances $\bar{H}_k(X_i^k),\ i = 1, \ldots, N_k$ from largest to smallest, and then taking the $\lceil (1-\rho_k)N_k \rceil$th order statistic. (*iii*) The threshold function should now be modified as

$$\widetilde{I}(x,\gamma) := \begin{cases} 0 & \text{if } x \ge \gamma + \varepsilon, \\ (\gamma + \varepsilon - x)/\varepsilon & \text{if } \gamma < x < \gamma + \varepsilon, \\ 1 & \text{if } x \le \gamma. \end{cases}$$

(iv) The inequalities at the beginning of steps 3 and $3b$ need to be replaced with $\widetilde{\gamma}_{k+1}(\rho_k, N_k) \le \bar{\gamma}_k - \varepsilon$ and $\widetilde{\gamma}_{k+1}(\bar{\rho}, N_k) \le \bar{\gamma}_k - \varepsilon$, respectively.

2. Similar to Chapter 5, a smoothed parameter updating procedure (cf. e.g., [26], [66]) is also used in actual implementation of the algorithm, i.e., first a smoothed parameter vector $\widehat{\theta}_{k+1}$ is computed at each iteration k according to

$$\widehat{\theta}_{k+1} := \upsilon\, \theta_{k+1} + (1-\upsilon)\widehat{\theta}_k, \quad \forall\ k = 0, 1, \ldots, \text{ and } \widehat{\theta}_0 := \theta_0,$$

where θ_{k+1} is the parameter vector derived at step 3 of SMRAS, and $\upsilon \in (0,1]$ is the smoothing parameter, then $f(x, \widehat{\theta}_{k+1})$ (instead of $f(x, \theta_{k+1})$) is used in step 1 to generate new samples.

6.5.1 Continuous Optimization

For continuous problems, we use multivariate normal p.d.f's as the parameterized probabilistic model. Initially, a mean vector μ_0 and a covariance matrix Σ_0 are specified; then at each iteration of the algorithm, it is easy to see that the new parameters μ_{k+1} and Σ_{k+1} are updated according to the following recursive formula:

$$\mu_{k+1} = \frac{\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})X_i^k}{\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})},$$

and

$$\Sigma_{k+1} = \frac{\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})(X_i^k - \mu_{k+1})(X_i^k - \mu_{k+1})^T}{\frac{1}{N_k}\sum_{i=1}^{N_k} \widetilde{S}(\bar{H}_k(X_i^k))\widetilde{I}(\bar{H}_k(X_i^k), \bar{\gamma}_{k+1})}.$$

By Corollary 6.4.2, the sequence of mean vectors $\{\mu_k\}$ will converge to the optimal solution x^* and the sequence of covariance matrices $\{\Sigma_k\}$ to the zero matrix. In subsequent numerical experiments, μ_{k+1} will be used to represent the best sample solution found at iteration k.

- Global Convergence

To demonstrate the global convergence of the proposed method, we consider the following two muti-extremal test functions

(1) Goldstein-Price function with additive noise

$$\begin{aligned} H_1(x,\psi) = \; & (1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)) \\ & (30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2))+\psi, \end{aligned}$$

where $x=(x_1,x_2)^T$, and ψ is normally distributed with mean 0 and variance 100. The function $h_1(x)=E_\psi[H_1(x,\psi)]$ has four local minima and a global minimum $h_1(0,-1)=3$.

(2) A 5-dimensional Rosenbrock function with additive noise

$$H_2(x,\psi)=\sum_{i=1}^{4}100(x_{i+1}-x_i^2)^2+(x_i-1)^2+1+\psi,$$

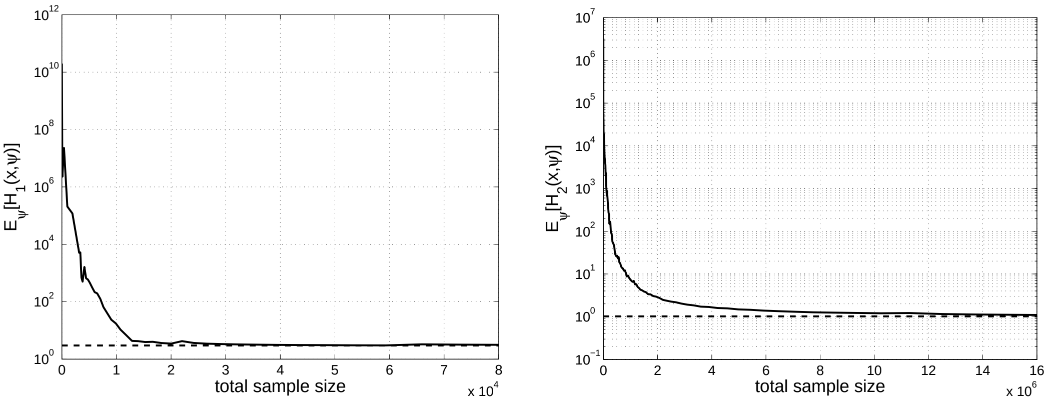
where $x=(x_1,\ldots,x_5)^T$, and ψ is normally distributed with mean 0 and variance 100. Its deterministic counterpart $h_2(x)=E_\psi[H_2(x,\psi)]$ has the reputation of being difficult to minimize and is widely used to test the performance of different global optimization algorithms. The function has a global minimum $h_2(1,1,1,1,1)=1$.

For both problems, the same set of parameters are used to test SMRAS: $\beta=1.02$, $\varepsilon=0.1$, mixing coefficient $\lambda_k=\frac{1}{\sqrt{k+1}}$ $\forall$ k, initial sample size $N_0=100$, $\rho_0=0.9$, $\alpha=1.03$, and the observation allocation rule is $M_k=1.1^k$, the stopping control parameters $\tau=0.005$ and $l=10$, the smoothing parameter $\upsilon=0.2$, the initial mean vector μ_0 is taken to be a n-by-1 vector of all 10's and Σ_0 is initialized as a n-by-n diagonal matrix with all diagonal elements equal to 100.

For each function, we performed 50 independent simulation runs of SMRAS. The averaged performance of the algorithm is shown in Table 6.1, where N_{avg} is the average total number of function evaluations needed to satisfy the stopping criteria, H_* and H^* are the worst and best function values obtained in 50 trials, and $\bar{H}$ is the averaged function values over the 50 replications. In Figure 6.2, we also plotted the average function values of the current best sample solutions for (a) function H_1 after 45 iteration of SMRAS, (b) function H_2 after 100 iterations of SMRAS.

H_i	$N_{avg}(std\ err)$	H_*	H^*	$\bar{H}(std\ err)$
H_1	5.40e+04(3.88e+02)	3.05	3.00	3.01(1.64e-3)
H_2	1.00e+07(4.92e+05)	1.31	1.02	1.09(9.10e-3)

Table 6.1: Performance of SMRAS on two test functions, based on 50 independent simulation runs. The standard errors are in parentheses.



(a) (b)

Figure 6.2: Performance of SMRAS on (a) Goldstein-price function; (b) 5-D Rosenbrock function.

- An Inventory Control Example

To further illustrate the algorithm, we consider an (s, S) inventory control problem with i.i.d. exponentially distributed continuous demands, zero order lead times, full backlogging of orders, and linear ordering, holding and shortage costs. The inventory level is periodically reviewed, and an order is placed when the inventory position (on hand plus that on order) falls below the level s, and the amount of the order is the difference between S and the current inventory position. Formally, we let D_t denote the demand in period t, X_t the inventory position in period t, p the per period per unit demand lost penalty cost, h the per period per unit inventory holding cost, c the per unit ordering cost, and K the set-up cost per order. The inventory position $\{X_t\}$ evolves according to the following dynamics

$$X_{t+1} = \begin{cases} S - D_{t+1} & X_t < s, \\ X_t - D_{t+1} & X_t \geq s. \end{cases}$$

The goal is to choose the thresholds s and S such that the long-run average cost per period is minimized, i.e.,

$$(s^*, S^*) = \arg\min J(s,S) := \arg\min \lim_{t\to\infty} J_t(s,S),$$

where $J_t(s,S) := \frac{1}{t}\sum_{i=1}^{t}\left[I\{X_i < s\}(K + c(S - X_i)) + hX_i^+ + pX_i^-\right]$, $I\{\cdot\}$ is the indicator function, $x^+ = \max(0,x)$, and $x^- = \max(0,-x)$. Note that the above objective cost function is convex; however, we will not exploit this property in our method. The primary reason we choose this problem as our test example is because its analytical optimal solution can be easily calculated (cf. e.g., [48]).

The following eight test cases, taken from [31], are used to test the performance of SMRAS. The cost coefficients and the optimal solutions are given in Table 6.2, each with $c = h = 1$ and exponentially distributed demands with mean $E[D]$.

In our simulation experiments, the initial mean vector is taken to be $(2000, 4000)^T$ for all eight cases, and the covariance matrices are initialized as diagonal matrices with all diagonal elements equal to 10^5 for cases $1-4$ and 10^6 for cases $5-8$. The other

Case	$E[D]$	p	K	J^*	s^*	S^*
1	200	10	100	740.9	341	541
2	200	10	10000	2200.0	0	2000
3	200	100	100	1184.4	784	984
4	200	100	10000	2643.4	443	2443
5	5000	10	100	17078	11078	12078
6	5000	10	10000	21496	6496	16496
7	5000	100	100	28164	22164	23164
8	5000	100	10000	32583	17582	27582

Table 6.2: The eight test cases.

parameters are: $\beta = 1.05$, $\varepsilon = 0.1$, $\lambda_k = \frac{1}{\sqrt{k+1}}$ $\forall$ k, $N_0 = 100$, $\rho_0 = 0.95$, $\alpha = 1.05$, $M_k = 1.2^k$, smoothing parameter $\upsilon = 0.3$. The average cost per period is estimated by averaging the accumulated cost over 50 periods after a warm-up length of 50 periods.

Figure 6.3 shows the typical performance of SMRAS for the first four test cases when the total number of simulation periods is set to 10^6. The locations of the optimal solutions are marked by $\bigstar$. We see that the algorithm converges rapidly to the neighborhood of the optimal solution in the first few iterations and then spends most of the computational effort in that small region. Numerical results for all eight test cases are given in Table 6.3. In the table, N_p indicates the total number of periods (including the warm-up periods) simulated, and the entries represent the averaged function values J of the final sample solutions obtained for different choices of N_p, each one based on 25 independent simulation replications.

Case	$N_p = 10^5$	$N_p = 10^6$	$N_p = 5 \times 10^6$	$N_p = 10^7$	J^*
1	1169.7(43.5)	742.6(0.32)	741.6(0.14)	741.2(0.06)	740.9
2	2371.6(37.8)	2223.9(3.57)	2202.0(0.20)	2200.8(0.17)	2200.0
3	1413.1(28.0)	1213.8(5.90)	1188.8(0.78)	1185.8(0.28)	1184.4
4	2709.0(13.4)	2667.2(4.89)	2647.2(0.61)	2645.0(0.42)	2643.4
5	18694.6(195.5)	17390.4(48.5)	17245.5(32.81)	17119.3(9.25)	17078
6	24001.7(340.8)	21808.5(53.6)	21780.0(34.00)	21520.9(5.80)	21496
7	32909.1(579.5)	28778.5(82.2)	28598.8(50.25)	28290.1(33.45)	28164
8	36520.0(538.0)	32881.7(216.9)	32860.2(52.56)	32682.8(36.68)	32583

Table 6.3: Performance of SMRAS on eight test cases, each one based on 25 independent simulation runs. The standard errors are in parentheses.

6.5.2 Combinatorial Optimization

To illustrate the performance of SMRAS on discrete stochastic optimization problems, we consider the buffer allocation problem in a service facility with unreliable servers. The system consists of m servers in series, which are separated by $m-1$ buffer locations. Each job enters the system from the first server, goes through all intermediate servers and buffer locations in a sequential order, and finally exits from the last server. The service times at each server are independent exponentially distributed with service rate $\mu_i,\ i = 1, \ldots, m$. The servers are assumed to be unreliable, and are subject to random failures. When a server fails, it has to be repaired. The time to failure and the time for repair are both i.i.d. exponentially distributed with respective rates f_i and $r_i,\ i = 1, \ldots, m$. A server is blocked when the buffer associated with the server coming next to it is full and is starved when no jobs are offered to it. Thus, the status of a server (busy/broken) will

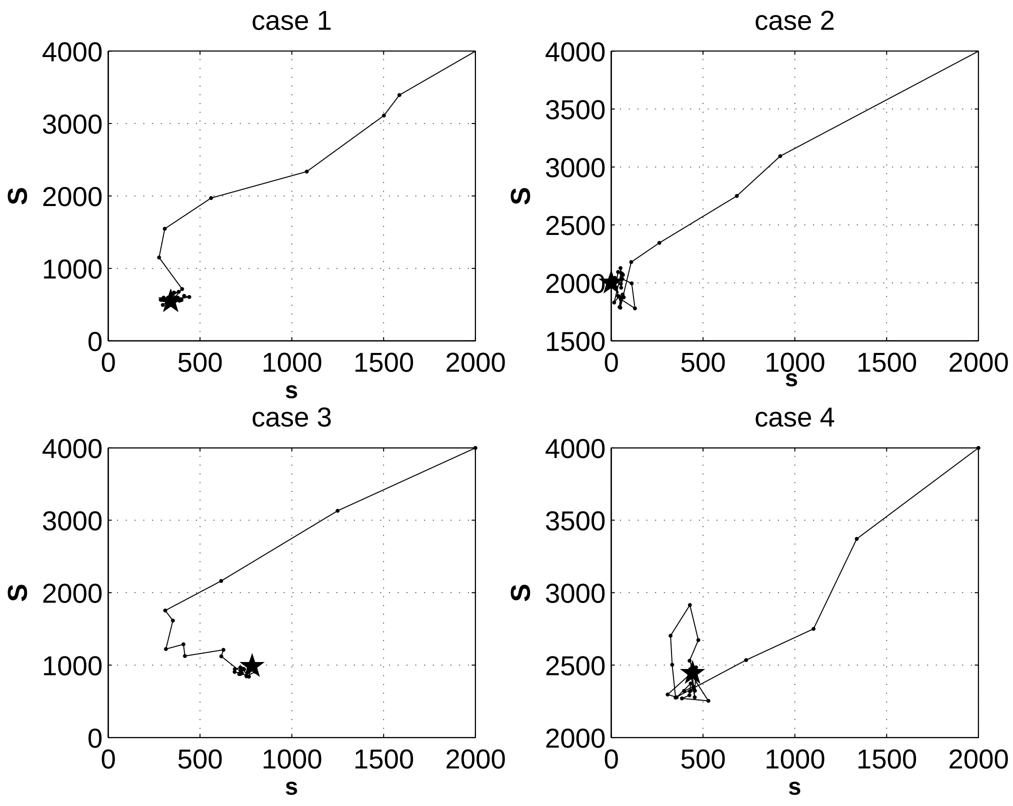


Figure 6.3: Typical performance of SMRAS on the first four test cases ($N_p = 10^6$).

affect the status of all other servers in the system. Figure 6.4 shows the four-server case, where server S_2 fails, which causes server S_1 to become blocked and server S_3 to become starved. We assume that the failure rate of each server remains the same, regardless of its current status. Given n limited buffer spaces, our goal is to find an optimal way of allocating these n spaces to the $m - 1$ buffer locations such that the throughput (average production rate) is maximized.

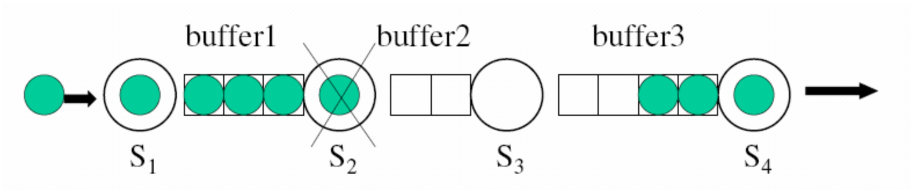


Figure 6.4: Graphical illustration of the buffer allocation problem.

When applying SMRAS, we have used the same technique as in [3] to generate admissible buffer allocations; the basic idea is to choose the probabilistic model as an $(n+1)$-by-$(m-1)$ matrix P, whose (i,j)th entry specifies the probability of allocating $i-1$ buffer spaces to the jth buffer location. Please refer to their paper for a detailed discussion. Once the admissible allocations are generated, it is straightforward to see that the entries of the matrix P are updated at the kth iteration as

$$P_{i,j}^{k+1} = \frac{\sum_{l=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_l^k))\widetilde{I}(\bar{H}_k(X_l^k), \bar{\gamma}_{k+1}) I\{X_{l,i}^k = j\}}{\sum_{l=1}^{N_k} \widetilde{S}_k(\bar{H}_k(X_l^k))\widetilde{I}(\bar{H}_k(X_l^k), \bar{\gamma}_{k+1})},$$

where X_l^k, $l = 1, \ldots, N_k$ are the N_k admissible buffer allocations generated, $\bar{H}_k(X_l^k)$ is the average throughput obtained via simulation when the allocation X_l^k is used, and $X_{l,i}^k = j$ indicates the event that j buffer spaces are allocated to the ith buffer location (i.e., the ith element of the vector X_l^k is equal to j).

For the numerical experiments, we consider two cases: (i) $m = 3$, $n = 1, \ldots, 10$, $\mu_1 = 1$, $\mu_2 = 1.2$ $\mu_3 = 1.4$, failure rates $f_i = 0.05$ and repair rates $r_i = 0.5$ for all $i = 1, 2, 3$; (ii) $m = 5$, $n = 1, \ldots, 10$, $\mu_1 = 1$, $\mu_2 = 1.1$, $\mu_3 = 1.2$, $\mu_4 = 1.3$, $\mu_5 = 1.5$, $f_i = 0.05$ and $r_i = 0.5$ for all $i = 1, \ldots, 5$.

Apart from their combinatorial nature, an additional difficulty in solving these problems is that different buffer allocation schemes (samples) have similar performances. Thus, when only noisy observations are available, it could be very difficult to discern the best allocation from a set of candidate allocation schemes. Because of this, in SMRAS we choose the performance function $S(\cdot)$ as an exponential function with a relatively larger base $\beta = 10$. The other parameters are as follows: $\varepsilon = 0.001$, $\lambda_k = 0.01$ $\forall$ k, initial sample size $N_0 = 10$ for case (i) and $N_0 = 20$ for case (ii), $\rho = 0.9$, $\alpha = 1.2$, observation allocation rule $M_k = (1.5)^k$, the stopping control parameters $\tau = 1e-4$ and $l = 5$, smoothing parameter $\upsilon = 0.7$, and the initial P^0 is taken to be a uniform matrix with each column sum equal to one, i.e., $P^0_{i,j} = \frac{1}{n+1}\ \forall\ i,\ j$. We start all simulation replications with the system empty. The steady-state throughputs are simulated after 100 warm-up events, and then averaged over the subsequent 900 events. Note that we have employed the sample reuse procedure (cf. Remark 6.3.1) in actual implementation of the algorithm.

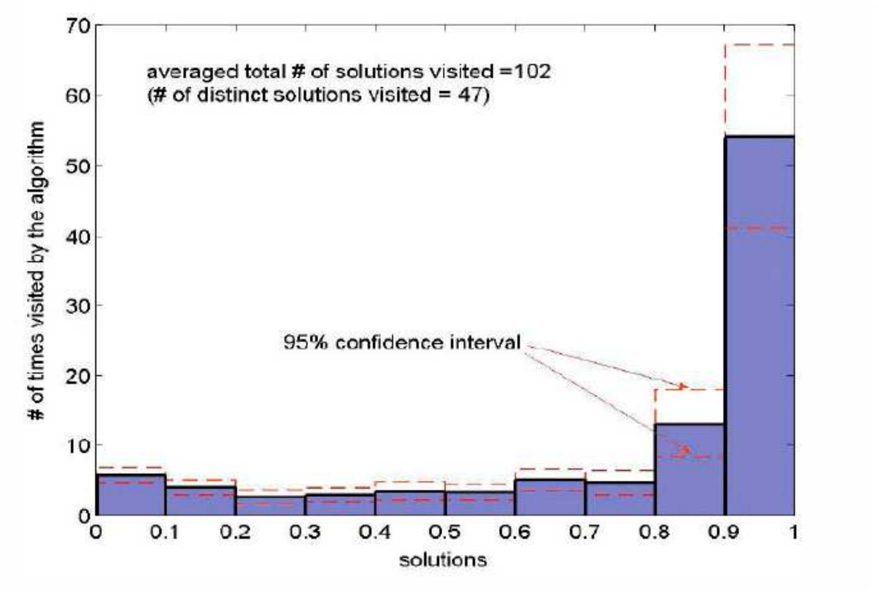


Figure 6.5: Performance of SMRAS on the buffer allocation problem (five-server $n = 10$ case).

Tables 6.4 and 6.5 give the performances of SMRAS for each of the respective cases (i) and (ii). In each table, N_{avg} is the averaged number of simulations over 16 independent trials, *Alloc* is the best allocation scheme and N_{A^*} is the number of times the best allocation found out of 16 runs, $\bar{\mathcal{T}}$ is the averaged throughput value calculated by the algorithm, and $\mathcal{T}^*$ represents the exact optimal solution (cf. [84]). We see that in both cases, SMRAS produces very accurate solutions while using only a small number of observations. To illustrate how SMRAS performs on this problem, we consider the five-server $n = 10$ case, where the total number of admissible allocation rules is 286. For the 286 solutions, we rank them from the worst to the best in terms of their performance

n	$N_{avg}(std\ err)$	$Alloc\ (N_{A^*})$	$\bar{\mathcal{T}}(std\ err)$	$\mathcal{T}^*$
1	33.1(0.49)	[1,0] (16)	0.634(4.06e-4)	0.634
2	46.8(3.15)	[1,1] (16)	0.674(6.35e-4)	0.674
3	43.9(1.51)	[2,1] (16)	0.711(6.11e-4)	0.711
4	49.8(3.45)	[3,1] (14)	0.735(6.47e-4)	0.736
5	50.4(3.68)	[3,2] (13)	0.758(1.06e-3)	0.759
6	64.0(6.29)	[4,2] (12)	0.776(1.39e-3)	0.778
7	59.1(4.27)	[5,2] (14)	0.792(1.04e-3)	0.792
8	63.9(4.79)	[5,3] (10)	0.805(1.20e-3)	0.806
9	60.6(3.46)	[6,3] (10)	0.817(6.53e-4)	0.818
10	63.7(5.69)	[7,3] (12)	0.826(9.88e-4)	0.827

Table 6.4: Performance of SMRAS on the buffer allocation problems case (i), based on 16 independent simulation runs. The standard errors are in parentheses.

and then equally partition these solutions into ten groups. For example, in Figure 6.5, the interval $[0, 1]$ represents the entire solution space, the interval $[0, 0.1]$ represents the worst 10% solutions, and $[0.9, 1]$ represents the top 10% best solutions. For SMRAS, the averaged total number of solutions visited is 102. Figure 6.5 shows that among the total 102 visits, the number of times each part of the solution space has been visited, where the red dashed line represents the 95% confidence interval. Obviously, we see that the best top 10% solutions have been visited significantly more often than solutions in other parts of the solution space. Also note that during the search of the algorithm, some solutions may be visited for a multiple number of times, the actually distinct number of solutions visited is only 47, only a small fraction of the solution space.

n	$N_{avg}(std\ err)$	$Alloc\ (N_{A^*})$	$\bar{\mathcal{T}}(std\ err)$	$\mathcal{T}^*$
1	1.02e+2(7.49)	[0,1,0,0] (16)	0.523(6.79e-4)	0.521
2	1.29e+2(14.8)	[1,1,0,0] (16)	0.555(3.86e-4)	0.551
3	1.75e+2(15.7)	[1,1,1,0] (16)	0.587(4.57e-4)	0.582
4	2.51e+2(25.9)	[1,2,1,0] (11)	0.606(1.20e-3)	0.603
5	3.37e+2(42.0)	[2,2,1,0] (10)	0.626(6.57e-4)	0.621
6	4.69e+2(55.2)	[2,2,1,1] (8)	0.644(1.10e-3)	0.642
7	4.56e+2(58.2)	[2,2,2,1] (7)	0.659(1.10e-3)	0.659
8	4.45e+2(54.9)	[3,2,2,1] (7)	0.674(1.10e-3)	0.674
9	5.91e+2(56.1)	[3,3,2,1] (6)	0.689(1.39e-3)	0.689
10	5.29e+2(54.0)	[3,3,3,1] (8)	0.701(1.10e-3)	0.701

Table 6.5: Performance of SMRAS on the buffer allocation problem case (*ii*), based on 16 independent simulation runs. The standard errors are in parentheses.

6.6 Conclusions

We have proposed a new randomized search method, called Stochastic Model Reference Adaptive Search (SMRAS), for solving both continuous and discrete stochastic global optimization problems. The method is shown to converge asymptotically to the optimal solution with probability one. The algorithm is general, requires only a few mild regularity conditions on the underlying problem; and thus can be applied to a wide range of problems with little modification. More importantly, we believe that the idea behind SMRAS offers a general framework for stochastic global optimization, based on which one can possibly design and implement other efficient algorithms.

There are several input parameters in SMRAS. In our preliminary numerical experiments, the choices of these parameters are based on trial and error. For a given problem, how to determine a priori the most appropriate values of these parameters is an open issue. One research topic is to study the effects of these parameters on the performance of the method, and possibly design an adaptive scheme to choose these parameters adaptively during the search process.

Our current numerical study with the algorithm shows that the objective function need not be evaluated very accurately during the initial search phase. Instead, it is sufficient to provide the algorithm with a rough idea where the good solutions are located. This has motivated our research to use observation allocation rules with adaptive increasing rates during different search phases. For instance, during the initial search phase, we could increase M_k at a linear rate or even keep it at a constant value; and exponential rates will only be used during the later search phase when more accurate estimates of the objective function values are required.

Some other research topics that would further enhance of the performance of SM-RAS include incorporating local search techniques in the algorithm and implementing a paralleled version of the method.

Chapter 7

Conclusions and Future Research

This dissertation consists of two main parts. The first part focuses on the development of new computational methodologies for solving Markov Decision Processes, where we have proposed two algorithms. The first algorithm is motivated by the computational challenges arising from settings where some of the parameters of the MDP models are either unknown or cannot be obtained in a feasible way. In particular, we have assumed that the underlying system can be simulated, and proposed to use multi-armed bandit models as efficient tools to adaptively allocate simulation samples to find good policies and/or value function estimates. We have shown the asymptotic unbiasedness of our approach, developed a convergence rate result, and studied its computational complexity. The second algorithm complements current existing state space reduction techniques, and addresses the solution of MDPs with large or uncountable action spaces. We have used an evolutionary population-based approach, which combines the specialized MDP solution techniques with ideas from evolutionary algorithms for optimization, to avoid carrying out an optimization over the entire action space. The convergence of the resultant algorithm is proved, and computational complexity is discussed. We have also compared the performance of our algorithm with those of other solution methods, including the classical policy iteration method and a recently proposed algorithm called evolutionary policy iteration. Numerical results demonstrate great promise of the proposed algorithm.

In the second part of this thesis, we have proposed a new randomized search (simulation-based) framework for solving general global optimization problems with little structure. The framework successfully addresses two of the most commonly encountered difficulties for many model-based search techniques, i.e., the problem of how to generate random samples and the problem of how to efficiently update probabilistic models. We argue that our framework can be easily used to construct a class of randomized global optimization algorithms with theoretical performance guarantee. Moreover, within this framework, the convergence analysis and practical performance of different algorithm instantiations will depend heavily on a sequence of independently constructed models called reference models. Thus, when constructing different instantiations, we can concentrate our effort on the design of these reference models. We have provided a particular instantiation of the framework, analyzed its convergence properties, and carried out detail numerical experiments to compare its performance with those of some other well-known methods like the Cross-Entropy method and simulated annealing. Both theoretical and empirical results demonstrate great potential of the proposed approach. In the final part of this thesis, we have rigorously discussed how to extend this framework to stochastic global optimization problems. Again, our discussion has been mostly centered around a particular algorithm instantiation, but we note that our work can be easily carried over to other various instantiations.

7.1 Future Work

This research has initiated some new and promising ideas in the field of decision making under uncertainty. However, there are still many refinements that can be explored. Some possible future research topics are outlined as follows.

In Chapter 3, we have proposed to use the multi-armed bandit model of [8] to adaptively choose which action to sample at each decision epoch, so that the resulting algorithm achieves logarithmic regret uniformly over time. However, this particular sampling strategy only gives us the asymptotic unbiasedness of the algorithm, a much weaker result than (almost sure) convergence. In this respect, it could be more useful to view the adaptive multi-stage sampling method as a simulation-based framework for solving finite-horizon MDPs, and look for different bandit models or even other different sampling techniques, so that it is possible for us to show stronger (almost sure) convergence of the resultant algorithms. Along this line, one possibility is to use the model reference adaptive search (MRAS) proposed in Chapter 5 as a potential sampling technique, and combine it with the AMS framework to yield yet another adaptive sampling algorithm. An additional advantage of using MRAS is that the finite-action-space assumption in the original AMS algorithm can be relaxed; the action space can be infinite or even uncountable.

When constructing sub-MDPs in ERPS, the action selection distribution $\mathcal{P}$ is currently held fixed throughout the entire search process. As discussed in Chapter 4.7, one possible and important line of research is to update the underlying action selection distribution based on the past sampling information so that more promising actions will have larger probabilities of being sampled in the future. Again, we believe that MRAS could be served as a promising candidate for updating these distributions. Thus, by combining MRAS with the so-called PICS step, it is possible to construct a new algorithm with balanced explorative and exploitative search that could be even more efficient in practice. Moreover, as mentioned in Chapter 4.7, there is no need to carry out an explicit local search at each iteration of the algorithm, since the sequence of action selection distributions will be getting more and more concentrated on regions containing high quality solutions (actions).

Regarding MRAS, we believe that there are several interesting future research directions. The most obvious one is perhaps to explore its potential applications in solving MDPs. This can be done either directly in the sense of [58], [68], where MDPs are interpreted as optimization problems over the policy spaces, or indirectly along the lines we just discussed in the previous two paragraphs. Another important direction is to study the convergence rate and the computational complexity of MRAS, perhaps for a class of problems (e.g., Lipschitz continuous, convex problems) of interest. The work of [74] and [89] in annealing adaptive search (AAS) (which involves the use of Boltzmann distributions) sheds some light in this area. Thus, one possibility, in particular, is to investigate the use of the Boltzmann distributions as the reference distributions in MRAS, and see if some nice properties (including convergence, rate, and complexity in the context of AAS) of the Boltzmann distributions are preserved by the method. From a more general point of view, we can always construct reference models that exploit the structures of the underlying problems, and thus design algorithms tailored to particular applications. The third direction is to develop new convergent algorithm instantiations, but with only fixed (sample) population size, perhaps via the use of past sampling information. This is especially attractive in the context of stochastic optimization where the simulation/observation cost is expensive, since the current version of MRAS requires the population size to increase in order to guarantee theoretical convergence.

BIBLIOGRAPHY

[1] Agrawal, R., "Sample mean based index policies with $O(\log n)$ regret for the multi-armed bandit problem," *Advances in Applied Probability,* **27**, 1054–1078 (1995).

[2] Agrawal, R., Teneketzis, D., and Anantharam, V., "Asymptotically efficient adaptive allocation schemes for controlled Markov chains: finite parameter space," *IEEE Trans. on Automatic Control,* **34**, 1249–1259 (1989).

[3] Allon, G., Kroese, D. P., Raviv, T., and Rubinstein, R. Y., "Application of the cross-entropy method to the buffer alloation problem in a simulation-based environment," *Annals of Operations Research,* **134**, 137–151 (2005).

[4] Alrefaei, M. H., and Andradóttir, S., "A modification of the stochastic ruler method for discrete stochastic optimization," *European Journal of Operational Research,* **133**, 160–182 (1995).

[5] Alrefaei, M. H., and Andradóttir, S., "A simulated annealing algorithm with constant temperature for discrete stochastic optimization," *Management Science,* **45**, 748–764 (1999).

[6] Andradóttir, S., "A method for discrete stochastic optimization," *Management Science,* **41**, 1946–1961 (1996).

[7] Andradóttir, S., "A global search method for discrete stochastic optimization," *SIAM Journal on Optimization,* **6**, 513–530 (1996).

[8] Auer, P., Cesa-Bianchi, N., and Fisher, P., "Finite-time analysis of the multiarmed bandit problem," *Machine Learning,* **47**, 235–256 (2002).

[9] Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E., "The nonstochastic multiarmed bandit problem," *SIAM J. Comput.* **32**, 48-77 (2002).

[10] Barash, D., "A genetic search in policy space for solving Markov decision processes," *AAAI Spring Symposium on Search Techniques for Problem Solving under Uncertainty and Incomplete Information,* Stanford University (1999).

[11] Bellman, R., Kalaba, R, and Kotkin, B., "Polynomial approximation – a new computational technique in dynamic programming: allocation processes," *Mathematics of Computation,* **17**, 82 (1963).

[12] Bentley, J., "Multidimensional binary search trees in database applications," *IEEE Trans. on Software Engineering,* **5**, 333–340 (1979).

[13] Bertsekas, D. P. *Dynamic Programming and Optimal Control* Volumes 1 and 2, Athena Scientific, Belmont, MA, 1995.

[14] Bertsekas, D. P. "Differential training of rollout policies," *Proc. 35th Allerton Conference on Communication, Control, and Computing,* Allerton Park, IL, 913-922 (1997).

[15] Bertsekas, D. P. and Castañon, D. A., "Adaptive aggregation methods for infinite horizon dynamic programming," *IEEE Trans. on Automatic Control,* **34**, 589–598 (1989).

[16] Blondel, V. D., and Tsitsiklis, J., "A survey of computational complexity results in systems and control," *Automatica,* **36**, 1249–1274 (2000).

[17] Broadie, M., and Glasserman, P., "Pricing American-Style securities using simulation," *Journal of Economic Dynamics and Control,* **21**, 1323–1352 (1997).

[18] Cesa-Bianchi, N., and Fisher, P., "Finite-time regret bounds for the multiarmed bandit problem," *Proc. 15th Int. Conf. on Machine Learning,* Morgan Kaufmann Publishers, San Francisco, CA 101-108 1998.

[19] Chang, H. S., Lee, H. G., Fu, M. C., and Marcus, S. I., "Evolutionary policy iteration for solving Markov decision processes," *IEEE Trans. on Automatic Control,* to appear (2006).

[20] Chang, H. S., Fu, M. C., Hu, J., and Marcus, S. I., "An asymptotically efficient simulation-based algorithm for finite horizon stochastic dynamic programming," *IEEE Trans. on Automatic Control,* accepted, (2006).

[21] Chang, H. S., Givan, R. L., and Chong, E. K. P., "Parallel Rollout for Online Solution of Partially observable Markov decision processes," *Discrete Event Dynamic Systems: Theory and Application,* **14**(3), 309-341 (2004).

[22] Chang, H. S., Fu, M. C., Hu, J., and Marcus, S. I., "An adaptive sampling algorithm for solving Markov decision processes." *Operations Research,* **51**(1), 126–139, (2005).

[23] Chavez, E., and Navarro, G., "An effective clustering algorithm to index high dimensional metric spaces," *Seventh International Symposium on String Processing Information Retrieval (SPIRE'00),* A Coru`na, Spain, pp. 75 (2000).

[24] Corana, A., Marchesi M., Martini, C., and Ridella, S., "Minimizing multimodal functions of continuous variables with the "Simulated Annealing" algorithm," *ACM Trans. on Mathematical Software,* **13**(3) 262–280 (1987).

[25] Cormen, T. H., Leiserson, C. E., and Rivest, R. L., *Introduction to Algorithms* MIT Press, Cambridge, MA 1990.

[26] De Boer, P. T., Kroese, D. P., Mannor, S., and Rubinstein, R. Y., “A tutorial on the cross-entropy method,” *Annals of Operation Research,* **134**, 19-67 (2005).

[27] de Farias, D. P. and Van Roy, B., “The linear programming approach to approximate dynamic programming,” *Operations Research,* **51**(6) 850–865 (2003).

[28] Demmel, J. W., *Applied Numerical Linear Algebra,* Soc. for Indust. and Appl. Math., Philadelphia, PA, 1997.

[29] Dorigo, M., and Gambardella, L. M., “Ant colony system: a cooperative learning approach to the traveling salesman problem,” *IEEE Trans. on Evolutionary Computation,* **1**, 53–66 (1997).

[30] Even-Dar, E., Mannor, S. and Mansour, Y., “Action elimination and stopping conditions for reinforcement learning,” *International Conference on Machine Learning,* pages 162–169 (2003).

[31] Fu, M. C., and Healy, K. J., “Techniques for simulation optimization: an experimental study on an (s, S) inventory system,” *IIE Transactions,* **29**, 191–199, (1997).

[32] Fu, M. C., *Gradient Estimation,* Chapter 19 in Handbooks in Operations Research and Management Science: Simulation, S.G. Henderson and B.L. Nelson, eds., Elsevier, 2006.

[33] Fu, M. C. and Jin, X., “Convergence of sample path optimal policies for stochastic dynamic programming,” Technical Research Report, Institute for Systems Research, University of Maryland, TR2005-84 (2005).

[34] Glasserman, P., "Performance continuity and differentiability in Monte Carlo optimization," *Proceedings of the 1988 Winter Simulation Conference,* M. Abrams, P. Haigh, and J. Comfort (eds.) (1988).

[35] Glover, F., "Tabu search: a tutorial," *Interfaces,* **20**, 74–94, (1990).

[36] Graves, T.L., and Lai, T. L., "Asymptotically efficient adaptive choice of control laws in controlled Markov chains," *SIAM J. Control Optimization,* **35**, 715–743 (1997).

[37] Gutjahr, W. J. "A converging ACO algorithm for stochastic combinatorial optimization," *Proc. SAGA 2003 Stochastic Algorithms: Foundations and Applications,* Hatfield (UK), A. Albrecht, K. Steinhoefl, eds., Springer LNCS 2827 10-25 (2003).

[38] Guttman, A., "R-trees: a dynamic index structure for spatial searching," *In Proc. ACM SIGMOD'84*, 47–57 (1984).

[39] Hernández-Lerma, O., and Lasserre, J. B., "Error bounds for rolling horizon policies in discrete-time Markov control processes," *IEEE Trans. on Automatic Control,* **35**, 1118–1124 (1990).

[40] Hoeffding, W., "Probability inequalities for sums of bounded random variables," *Journal of the American Statistical Association,* **58**, 13–30 (1963).

[41] Homem-de-Mello, T., "A study on the cross-entropy method for rare event probability estimation," Technical Report 04-002, Department of Industrial Engineering and Management Sciences, Northwestern University (2004). http://users.iems.northwestern.edu/ tito/pubs/rarevents_submitted.pdf

[42] Hong, L. J., and Nelson, B. L., "Discrete optimization via simulation using COMPASS," *Operations Research,* forthcoming (2006).

[43] Hu, J., Fu, M. C., and Marcus, S. I., "Model reference adaptive search: a new approach to global optimization," *late-breaking paper, Genetic and Evolutionary Computation Conference (GECCO)*, Washington D. C. (2005).

[44] Hu, J., Fu, M. C., and Marcus, S. I., "Simulation optimization using model reference adaptive search," *Proceedings of the 2005 Winter Simulation Conference,* 811–818 (2005).

[45] Hu, J., Fu, M. C., Ramezani, V., and Marcus, S. I., "An evolutionary random policy search algorithm for solving Markov decision processes," *INFORMS Journal on Computing,* forthcoming (2006).

[46] Hu, J., Fu, M. C., and Marcus, S. I., "A model reference adaptive search method for global optimization," *Operations Research,* forthcoming (2006).

[47] Hu, J., Fu, M. C., and Marucs, S. I., "A model reference adaptive search method for stochastic global optimization," submitted for publication (2006).

[48] Karlin, S., *Steady State Solutions,* Studies in the Mathematical Theory of Inventory and Production. Arrow, K. J., Karlin, S., and Scarf H. (eds.), Stanford University Press (1958).

[49] Kearns, M., Mansour, Y., and Ng, A. Y., "A sparse sampling algorithm for near-optimal planning in large Markov decision processes," *Machine Learning,* **49**, 193–208 (2001).

[50] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by simulated annealing," *Science,* **220**, 671-680 (1983).

[51] Kroese, D. P., Rubinstein, R. Y., and Porotsky, S., "The cross-entropy method for continuous multi-extremal optimization," *Operations Research,* Under review (2005).

[52] Lai, T., and Robbins, H., "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics,* **6**, 4–22 (1985).

[53] Larrañaga, P., Etxeberria, R., Lozano, J. A., Sierra, B., Iñza, I., and Peña, J. M., "A review of the cooperation between evolutionary computation and probabilistic graphical models," *Proceedings of the Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF* 99. Special Session on Distributions and Evolutionary Computation, 314–324 (1999).

[54] Law, A. M., and Kelton, W.D., *Simulation Modeling and Analysis* 3rd ed. McGraw-Hill, New York, 2002.

[55] Lin, A. Z. -Z., Bean, J., and White, C. III, "A hybrid genetic/optimization algorithm for finite horizon partially observed Markov decision processes," Technical Report 98-25, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor (1998).

[56] Lourenco, H. R., Martin, O. C., and Stützle, T., *Iterated local search,* Handbook on MetaHeuristics. Ed. Glover, F., and Kochenberger, G. 321–353, Kluwer Academic Publishers, Norwell, MA, 2002.

[57] MacQueen, J., "A modified dynamic programming method for Markovian decision problems," *J. Math. Anal. Appl.,* **14** 38–43 (1966).

[58] Mannor, S., Rubinstein, R., and Gat, Y., "The cross-entropy method for fast policy search," *International Conference on Machine Learning,* 512–519 (2003).

[59] Mühlenbein, H., and Paaß, G., "From recombination of genes to the estimation of distributions: I. binary parameters," *In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, editors, Parallel Problem Solving from Nature - PPSN IV,* 178–187, Berlin, Springer Verlag, (1996).

[60] Pelikan, M., Goldberg, D. E., and Lobo, F. G., "A survey of optimization by building and using probabilistic models," Urbana, IL: University of Illinois Genetic Algorithms Laboratory (IlliGAL report No. 99018) (1999).

[61] Pintér, J. D., *Global Optimization in Action,* Kluwer Academic Publisher, The Netherlands, 1996.

[62] Puterman, M. L. and Shin, M. C., "Modified policy iteration algorithms for discounted Markov decision processes," *Management Science,* **24**, 1127–1137 (1978).

[63] Puterman, M. L., *Markov Decision Processes: Discrete Stochastic Dynamic Programming,* Wiley & Sons, New York 1994.

[64] Ross, S., *Stochastic Process* 2nd ed. John Wiley & Sons, 1995.

[65] Rubinstein, R. Y., "Optimization of computer simulation models with rare events," *European Journal of Operations Research,* **99**, 89–112 (1997).

[66] Rubinstein, R. Y., "The cross-entropy method for combinatorial and continuous optimization," *Methodology and Computing in Applied Probability,* **2**, 127–190 (1999).

[67] Rubinstein, R. Y., "Combinatorial optimization, ants and rare events," In S. Uryasev and P. M. Pardalos, editors, Stochastic Optimization: Algorithms and Applications, 304–358 Kluwer (2001).

[68] Rubinstein, R. Y., and Kroese, D. P., *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning,* Springer, New York 2004.

[69] Rubinstein, R. Y., and Shapiro, A., *Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method,* John Wiley & Sons 1993.

[70] Rust, J., "Structural estimation of Markov decision processes," *Engle, R. and McFadden, D. (eds.) Handbook of Econometrics,* North Holland: Amsterdam (1994).

[71] Rust, J., "Numerical dynamic programming in economics," *chapter 14 in Amman, H., Kendrick, D. and Rust J. (eds.) Handbook of Computational Economics,* Elsevier, North Holland (1996).

[72] Rust, J., "Using randomization to break the curse of Dimensionality," *Econometrica,* **65**(3) 487–516 (1997).

[73] Schweitzer, P. J., and Seidman, A., "Generalized polynomial approximations in Markovian decision problems," *J. Math. Anal. and Appl.* **110**, 568–582 (1985).

[74] Shen Y., *Annealing Adaptive Search with Hit-and-Run Sampling Methods for Global Optimization,* Ph.D. Thesis, Department of Industrial Engineering, University of Washington, Seattle 2005.

[75] Shi, L., and Ólafsson, S., "Nested partitions method for global optimization," *Operations Research,* **48**, 390–407, (2000).

[76] Shi, L., and Ólafsson, S., "Nested partitions method for stochastic optimization," *Methodology and Computing in Applied Probability,* **2**, 271–291 (2000).

[77] Shiryaev, A. N., *Probability,* Second Edition, Springer-Verlag, New York, 1995.

[78] Spall, J. C., "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Transactions on Automatic Control,* **37** 332–341 (1992).

[79] Srinivas, M., and Patnaik, L. M., "Genetic algorithms: a survey," *IEEE Comput.,* **27**(6) 17–26 (1994).

[80] Sutton, R. S., "Learning to predict by the method of temporal differences," *Machine Learning,* **3**, 9–44 (1988).

[81] Thomas, L. C., Hartley, R., and Lavercombe, A. C., "Computational comparison of value iteration algorithms for discounted Markov decision processes," *Operations Research Letters,* **2** 72–76 (1983).

[82] Trick, M. and Zin, S., "Spline approximations to value functions: a linear programming approach," *Macroeconomic Dynam.,* **1** 255-277 (1997).

[83] Tsitsiklis, J. N. and Van Roy, B., "Feature-based methods for large-scale dynamic programming," *Machine Learning,* **22** 59–94 (1996).

[84] Vouros, G. A., and Papadopoulos, H. T., "Buffer allocation in unreliable production lines using a knowledge based system," *Computer & Operations Research,* **25**, 1055–1067 (1998).

[85] Watkins, C. J. C. H., "Learning from delayed rewards," PhD thesis, University of Cambridge (1983).

[86] Wells, C., Lusena, C., and Goldsmith, J., "Genetic algorithms for approximating solutions to POMDPs," Department of Computer Science Technical Report TR-290-99, University of Kentucky (1999). http://cs.engr.uky.edu/ goldsmit/papers/gen.ps

[87] Yan, D., and Mukai, H., “Stochastic discrete optimization,” *SIAM Journal on Control and Optimization,* **30**, 594–612 (1992).

[88] Yao, X., and Liu, Y., “Fast evolutionary programming,” *Proceedings of the* 5*th Annual Conference on Evolutionary Programming,* 451–460, MIT Press (2000).

[89] Zabinsky, Z. B., *Stochastic Adaptive Search for Global Optimization,* Kluwer Academic Publisher, Norwell, MA 2003.

[90] Zhang, Q., and Mühlenbein, H., “On the convergence of a class of estimation of distribution algorithm,” *IEEE Trans. on Evolutionary Computation,* **8**, 127–136 (2004).

[91] Zlochin, M., Birattari, M., Meuleau, N., and Dorigo, M., “Model-based search for combinatorial optimization: a critical survey,” *Annals of Operations Research,* **131**, 373–395 (2004).