ADAPTIVE STOCHASTIC SIMULATION FOR STRUCTURED PROBLEMS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Samuel Ehrlichman

August 2008

ADAPTIVE STOCHASTIC SIMULATION FOR STRUCTURED PROBLEMS

Samuel Ehrlichman, Ph.D.

Cornell University 2008

In this thesis, I examine several situations in which one can improve the efficiency of a stochastic simulation algorithm by adaptively exploiting special structure of the problem at hand.

The thesis is comprised of three independent papers. In the first paper, I propose a new variance reduction technique in the setting of comparing the performance of two stochastic systems. The technique is a natural generalization of common random number sampling, a well-known sampling strategy that may reduce variance in certain situations. Common random number sampling entails sampling the underlying uniform random variates according to a particular copula; my proposed method considers more general copulae. I identify properties such a copula must have in order to induce a valid sampling strategy, give examples of situations in which copulae exist that outperform common random numbers, and give an algorithm for computing an effective Gaussian copula.

In the second paper, I discuss an automated procedure for computing a control variate for the pricing of American options. The control variate is equal to the value of a particular martingale at the time the option is exercised. The martingale is constructed using a an approximation of the value function in a backward dynamic program. We use adaptive linear regression splines to create the functional approximation. These splines have properties of which make them computationally convenient for constructing a martingale-based control variate of this type.

In the third paper, I describe and analyze a novel root-finding procedure for monotone, convex univariate functions in the presence of noise. The main result of the analysis is a probabilistic performance guarantee for the algorithm. Specifically, given an indifference parameter delta and a confidence alpha, the algorithm returns a point whose absolute function value is less than delta with probability at least one minus alpha. The total amount of work required by the algorithm may be bounded in terms of the length of the compact interval on which the function is defined and the Lipschitz constant of the function.

BIOGRAPHICAL SKETCH

Although Sam was born in the comparative tranquility of central New Jersey, he spent most of his childhood in Flushing, Queens. Flushing has many claims to fame: it is one of New York City's most ethnically diverse communities; it is often considered a birthplace of the religious freedom movement; and it is the location of that most malodorous river described in *The Great Gatsby*.

For six years Sam enjoyed a twice-daily hour commute by subway from Flushing to Hunter College High School. He then escaped the New York metropolitan area for a brief while, attending Swarthmore College from 1991 to 1995. At Swarthmore, Sam studied mathematics and computer science, did lots of a capella singing, and played way too much bridge.

Sam returned to Queens after college, taking a job in the financial software industry in Manhattan. In 1999, he founded a start-up company for providing risk analytics to individual investors over the Internet. His experience as an entrepreneur was marked by a certain elegant minimalism: the firm had no funding, no assets, no revenue, and no employees. Nevertheless, they were acquired in late 2000 by another start-up, one that at least had some funding. Two years later, as that ship began to sink in the lowering economic tides, Sam came to Cornell to study Operations Research.

After graduating from Cornell, Sam intends to fully enjoy his last summer in Ithaca: the Farmers' Market, the gorges, the Grassroots Festival, and all the wonderful friends he's made here. After that, it's back to work; Sam will start as a quantitative researcher at Jane Street Capital, LLC in August, 2008.

And he will again be living in New Jersey. Apparently, you can take the boy out of the Jerz, but you cannot take the Jerz out of the boy.

This document is dedicated to my wife Kari Tetzlaff, my daughter Cecilia Ehrlichman, and my parents Howard and Elizabeth Ehrlichman.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Shane G. Henderson, for his unwavering support. Shane has been a delight to work with. He has consistently guided my growth as a researcher, provided insightful and careful feedback on all my work, and helped me maintain confidence and focus in times of frustration. Shane has also been an invaluable resource for me in my role as a new teacher. All in all, one could not hope for a better advisor!

I would also like to thank the entire faculty here in the School of Operations Research and Information Engineering for creating such a welcoming and intellectually stimulating environment. Special thanks go to my committee members Bob Jarrow and David Ruppert for their thoughts on matters financial and statistical, to David Shmoys and Gena Samorodnitsky for recruiting me to come to Cornell, to Jim Renegar and Kathryn Caggiano for their support of my teaching, to Mike Todd for useful conversations about my research, to Philip Protter for many interesting discussions of finance and politics, to Stefan Weber for many interesting discussions of finance and life, and to Bob Bland for all-around support and encouragement. Extra special thanks go to fellow Flushingite Sid Resnick for being a worthy verbal sparring partner.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Background

Stochastic simulation is a widely used technique applicable to a great variety of computational problems in operations research. One reason for the success that simulation has enjoyed is the fact that the rate of convergence of a simulation estimator is typically independent of the dimension of the underlying problem, thanks to the Central Limit Theorem (e.g., Henderson 2006). Thus, for very high dimensional problems, simulation is often the most appropriate choice of computational method.

Perhaps an even more important reason for simulation's popularity is its simplicity. At its most basic level, stochastic simulation entails generating a sequence of random values and substituting the empirical distribution of these values for their theoretical distribution in computations. In many cases, this process is fairly straightforward. One particularly simple example is the estimation of the mean of a random variable's distribution when that distribution is not known analytically. If we can generate values of the random object $\mathbf{X}$ according to a known distribution μ, and if we can compute the function value $f(\mathbf{X})$ for any realization of $\mathbf{X}$, then we can compute the simulation estimator $\frac{1}{n}\sum_{j=1}^{n} f(\mathbf{X}_j)$ for $Ef(\mathbf{X})$.

The apparent simplicity of implementing a simulation estimator of the kind described above may lead one to believe that simulation is always a trivial exercise. The reality is that the devil is the details. Many challenges persist in computing simulation estimators. In some situations, the computational effort required to generate the random values $\mathbf{X}_j, j \geq 1$ or to compute the function f is too great for large numbers of iterations be practical. In these cases people often look for estimators having lower variance than the usual sample average estimator so that fewer iterations are necessary. Techniques for computing such estimators are known as variance reduction techniques (e.g., Asmussen and Glynn 2007, Chapter 5). Although reducing the variance of an estimator does not alter its rate of convergence, it does provide a constant factor improvement in speed.

Much current research in stochastic simulation focuses on simulation procedures that are adaptive in some sense. There is a bit of subjectivity involved in defining exactly what we mean by "adaptive." For the purposes of this dissertation, I take the expression to mean that certain features of the simulation algorithm are learned and adjusted by the algorithm as it proceeds. A crucial feature of adaptivity is that its details should be invisible to the end-user of the simulation algorithm.

What is it that adaptive simulation algorithms adapt to? They adapt to structure present in the problem at hand. Sometimes this structure is present but completely unknown, whereas in other cases a good deal is known about the problem *a priori*. The papers presented in this thesis contain examples of both types.

1.2 Comparing Two Systems: Beyond Common Random Numbers

The first paper of this thesis is Chapter 2, "Comparing Two Systems: Beyond Common Random Numbers." A version of this paper will appear in the Proceedings of the 2008 Winter Simulation Conference (Ehrlichman and Henderson

2008).

In the paper, I propose a new variance reduction technique applicable to the problem of comparing the real-valued output of two stochastic simulations. The technique is quite general in that very little structure about the systems being compared is assumed. Rather, all the structure is learned by the algorithm.

A classic variance reduction trick for the system comparison problem is common random number (CRN) sampling (e.g., Kelton 2006). If the outputs being compared are represented by random variables X and Y, the quantity of interest is $X - Y$. CRN sampling attempts to reduce the sampling variance not of X or Y individually but rather to reduce the sample variance of the difference $X - Y$ by inducing positive covariance between the estimators of X and Y. The idea is to hope that the random variables X and Y depend upon the underlying sequence of computer-generated pseudorandom variates in similar ways. In that case, feeding the same sequence of pseudorandom variates to the estimators of X and Y should result in the desired positive covariance.

Common random number sampling is only effective if the necessary similarity between the systems is present. In general, there is no reason to believe this will hold. In fact, it is possible that CRN sampling will introduce *negative* covariance between X and Y! See Wright and Ramsay (1979) for an example.

The contribution of our work is to describe a generalization of CRN sampling that adapts to the observed structure of the systems X and Y. Instead of using identical streams of random numbers for each system, our procedure discovers a joint distribution (copula) on the streams that minimizes the variance of $X - Y$ on a pilot sample. I provide several examples in which this procedure outperforms both naïve sampling and CRN sampling. I also prove a key property of the optimal copula in the special case where both X and Y are linear in a certain transformation of the pseudorandom streams.

1.3 Adaptive Control Variates for Multidimensional American Options

The second paper of this thesis is Chapter 3, "Adaptive Control Variates for Multidimensional American Options." This paper appears in *The Journal of Computational Finance* (Ehrlichman and Henderson 2007a).

This paper describes a variance reduction scheme for American option pricing. The scheme is to be used in conjunction with many well-known simulation-based American option pricing algorithms such as Longstaff and Schwartz (2001) or Tsitsiklis and Van Roy (2001). It can be applied to any option payoff structure and any underlying risk-neutral market dynamics provided that certain key conditional expectations are easily computed.

Variance reduction in this scheme is achieved by way of a control variate. The value of the control variate on any given sample path is equal to the value of a particular martingale at the exercise time of the option. The effectiveness of the control variate depends upon two features: the quality of the exercise strategy determined by the underlying pricing algorithm (e.g., Longstaff-Schwartz), and the quality of a certain approximation to the option's value function.

Use of martingales as control variates in this way is not new. In fact, Bolia and Juneja (2005) use just such a martingale for American option pricing. Their paper relies on a particular functional form for the approximate value function in the one-dimensional Black-Scholes case. Our contribution was to demonstrate an automatic method for constructing the martingale using multivariate linear regression splines. Because of the specific form these splines take, it turns out to be rather easy to compute the necessary conditional expectations in a great variety of cases.

The paper includes the results of extensive numerical tests of our method. The method is quite successful in practice, sometimes achieving variance reduction factors of more than 100.

1.4 Deterministic and Stochastic Root Finding in One Dimension for Increasing Convex Functions

The third paper of this thesis is Chapter 4, "Deterministic and Stochastic Root Finding in One Dimension for Increasing Convex Functions." As of the writing of this thesis, the paper was in review for publication in *SIAM Journal on Optimization.* An early version appeared in the Procedings of the 2007 Winter Simulation Conference (Ehrlichman and Henderson 2007b).

In this paper I prove a probabilistic performance guarantee for a novel root-finding algorithm. The algorithm is applicable to any non-decreasing, convex, univariate function on a compact interval having a known Lipschitz constant.

In analyzing this root-finding algorithm, I adopt an indifference-zone approach. The idea is that when seeking a root of a function h, we consider any point x having $|h(x)| < \delta$ to be "close enough." Such a point is called a δ-root of h. Indifference zones have their roots in the problem of constructing statistically valid sequential tests (e.g., Wald 1947). More recently, indifference zones have appeared in a related context in the simulation literature, the "selection of the best system" problem. See Kim and Nelson (2006).

I consider both the case where the function may be evaluated exactly and the case where only interval estimates are available. The algorithm evaluates the function at a sequence of points such that each point is adaptively chosen to guarantee a minimal reduction in the complexity of the problem. I prove that this strategy leads to an overall probabilistic performance guarantee for the algorithm. Specifically, given an indifference zone parameter $\delta > 0$ and a confidence level $\alpha \in (0, 1)$, the algorithm is guaranteed to produce a δ-root with probability at least $1 - \alpha$.

The original motivation for this paper came from the American option pricing problem, although in a more restricted setting than I consider in Chapter 3. In the one-dimensional Black-Scholes setting, it is optimal to exercise an American put at time $t < T$ if the stock price S_t is below a certain threshold. Here, T denotes the expiration time of the option. If $g(s)$ denotes the value of immediately exercising the option at time t when the stock price is s and $f(s)$ is the conditional expected value of holding the option past time t given that the stock price is s, then the desired threshold is equal to the unique root of $h = f - g$. It can be shown (e.g., Ekström 2004) that h is monotone and convex on the region where $g > 0$.

The indifference zone approach has a natural economic interpretation in the context of American options. Namely, $|h(\hat{x})|$ is an upper bound on the financial loss associated with choosing $\hat{x}$ as the exercise threshold in lieu of the unknown true root of h. Contrast this with the usual notion of closeness to a root, which is defined in terms of the abscissa.

I conclude this chapter by observing that the application of my root-finding algorithm to American option pricing is far from straightforward. The difficulty lies in the fact that although h is monotone and convex, we do not have access to the true value function f at any time step other than $t = T-1$. The *approximate* value function arising from following a suboptimal exercise strategy is, unfortunately, not convex. A useful direction for future research would be to complete the bridge between the root-finding algorithm and option pricing in such a way that the resulting option price admitted a probabilistic performance guarantee as well.

Chapter 2

Comparing Two Systems: Beyond Common Random Numbers

2.1 Introduction

Let X and Y be random variables quantifying the performance of two systems. Consider the problem of determining which of these systems has greater mean performance. A typical stochastic simulation approach to this problem is to generate an IID sequence $(X_j, Y_j : j = 1, \ldots, n)$ of pairs of random variables where $X_j \overset{d}{=} X$ and $Y_j \overset{d}{=} Y$, $j = 1, \ldots, n$, and estimate

$$E(X - Y) \approx \frac{1}{n} \sum_{j=1}^{n} (X_j - Y_j).$$

The sign of the resulting estimator indicates which of the two systems is preferable.

It is crucial that the random vectors (X_j, Y_j) be IID in order for the usual limit theorems to hold. However, there is no reason that X_j and Y_j must be independent for fixed j. Indeed, it may be helpful to induce such dependence; if $\operatorname{Cov}(X_j, Y_j) > 0$, then

$$\operatorname{Var}(X_j - Y_j) < \operatorname{Var} X + \operatorname{Var} Y.$$

The right-hand side, above, is the variance that would be achieved if X_j and Y_j were sampled independently.

A particularly simple method for inducing positive dependence between X and Y is common random number (CRN) sampling (e.g., Kelton 2006), discussed in §2.2. Our purpose in this paper is to propose a more general technique for introducing such dependence; the technique has a quite similar flavor to CRN sampling, and indeed has CRN sampling as a special case.

The outline of the paper is as follows. Section 2.2 establishes notation, defines CRN sampling, and introduces our new method. Section 2.3 discusses earlier work on the subject, especially on known conditions under which CRN is optimal (in a sense to be defined). Section 2.4 proposes using a particular class of copula, the Gaussian copula, and shows its effectiveness in several toy examples. Section 2.5 discusses two algorithms for computing an optimal Gaussian copula. In §2.6 we prove a key property of the set of optimal Gaussian copulae in a particularly simple case. Section 2.7 offers concluding remarks and directions for further research.

In the sequel, we drop the subscript j and make reference to the joint distribution of the random vector (X, Y).

2.2 Common Random Numbers

Common random number sampling entails using identical sequences $\mathbf{U}_X = \mathbf{U}_Y = \mathbf{U} = (U_1, U_2, \ldots)$ of pseudorandom variates to compute both X and Y. This can often be accomplished by resetting a seed for a pseudorandom number generator to a common value s for simulating both X and Y. If the ways in which X and Y are computed in terms of $\mathbf{U}$ are fairly similar, we may hope that this technique induces the positive dependence, and hence variance reduction, discussed in the previous section.

We shall ignore the fact that $\mathbf{U}$ is actually deterministic and treat it as random for the remainder of this paper. Under this convention, $\mathbf{U}$ is treated as a sequence of

IID uniform [0,1] random numbers. In fact, we will assume further that $\mathbf{U}$ has finite dimension d, i.e., $\mathbf{U} = (U_1, \ldots, U_d)$. Technically this is without loss of generality; in fact, we could even assume $d = 1$ since there exists a bijection between $[0, 1]$ and $[0, 1]^\infty$. But practically speaking, such a bijection is not particularly useful. Hence, the assumption of finite dimensionality may limit the situations in which the approach we discuss below is applicable.

Let us make the dependence of X and Y on $\mathbf{U}$ explicit by defining functions $f_U, g_U : [0, 1]^d \to \mathbf{R}$ so that $X = f_U(\mathbf{U}_X)$ and $Y = g_U(\mathbf{U}_Y)$, for $\mathbf{U}_X, \mathbf{U}_Y \sim \mathcal{U}([0, 1]^d)$. Notice that if X or Y depends on $j < d$ uniform random variables, then the function f_U or g_U will simply depend on the first j components of $\mathbf{U}_X$ or $\mathbf{U}_Y$.

Standard, or IID, sampling consists of sampling the random vector $(\mathbf{U}_X, \mathbf{U}_Y)$ according to the uniform probability measure on $[0, 1]^{2d}$. We denote this measure by P_{iid}. In contrast, CRN sampling consists of sampling under the probability measure P_{CRN} where

1. $\mathbf{U}_X$ and $\mathbf{U}_Y$ are each uniform on $[0, 1]^d$, and
2. $\mathbf{U}_X = \mathbf{U}_Y$ P_{CRN}-almost surely.

Both probability measures described above are examples of *copulae* on $[0, 1]^{2d}$. That is, they are distributions on this hypercube having uniform marginals. In this paper, we consider the possibility of using other copulae that satisfy Condition 1. We say that a copula on $[0, 1]^{2d}$ satisfying Condition 1 is *admissible*, and we denote by $\mathscr{C}$ the set of all such copulae. Within any given class of copulae, a copula minimizing $\operatorname{Var}(X - Y)$ is called *optimal* in that class. If $\operatorname{Var}(X - Y) = 0$ is achieved, the copula is called *perfect*. Of course, $\operatorname{Var} X$ and $\operatorname{Var} Y$ are unaffected by the choice of admissible copula. This implies that we cannot expect to find a perfect copula except possibly in the case where $\operatorname{Var} X = \operatorname{Var} Y$.

2.3 Previous Work

At its essence, our work involves a computational approach to constructing a coupling between two stochastic systems. For excellent reviews of coupling see Lindvall (1992) and Thorisson (2000). Wright and Ramsay (1979) describe a class of couplings between two (univariate) random variables based upon finite partitions of the unit interval; they (apparently erroneously) attribute the idea to Hammersley and Handscomb (1964) who in fact propose a related coupling in the context of generalized antithetic sampling. Schmeiser and Kachitvichyanukul (1986) described a number of approaches for coupling two random variables based on generation methods other than inversion. Devroye (1990) developed various couplings between two random vectors that attempts to maximize the number of components that are identical. Glasserman and Yao (1992) consider the question of when common random numbers is optimal for a class of performance measures that includes the variance of the difference between two random variables as considered here. As noted there, this question is poorly defined without further structure, which they impose in various ways. Glasserman and Yao (2004) provide a characterization of optimal couplings using a property they call the "nonintersection" property.

Throughout the remainder of the paper we will make considerable use of standard methods from linear algebra. The necessary background can be found in many books, e.g., Horn and Johnson (1985).

2.4 Gaussian Copulae

We now restrict our attention to a smaller class of copulae than $\mathscr{C}$, the class $\mathscr{G}$ of *Gaussian copulae* that are admissible. In our setting, a Gaussian copula on $[0,1]^{2d}$ is a probability measure P such that the random vector $(\mathbf{Z}_X, \mathbf{Z}_Y)$, defined componentwise by

$$Z_X[i] = \Phi^{-1}(U_X[i]),$$
$$Z_Y[i] = \Phi^{-1}(U_Y[i]),$$

for $i = 1, \ldots, d$, has a multivariate normal distribution with standard marginals under P. Here, Φ denotes the standard normal cdf.

Let

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_{XX} & \mathbf{\Sigma}_{XY} \\ \mathbf{\Sigma}_{XY}^T & \mathbf{\Sigma}_{YY} \end{bmatrix}$$

be the covariance matrix of $(\mathbf{Z}_X, \mathbf{Z}_Y)$, where the blocks are $d \times d$ matrices. In order for $\mathbf{U}_X$ and $\mathbf{U}_Y$ to be uniform on $[0,1]^d$ (Condition 1 of §2.2) we must have $\mathbf{\Sigma}_{XX} = \mathbf{\Sigma}_{YY} = \mathbf{I}_d$, the $d \times d$ identity matrix. Therefore we consider only covariance matrices of the form

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{I}_d & \mathbf{\Sigma}_{XY} \\ \mathbf{\Sigma}_{XY}^T & \mathbf{I}_d \end{bmatrix}. \tag{2.1}$$

A positive semidefinite matrix of the form (2.1) will be called *admissible*, and we denote by $\mathscr{S}_d$ the set of all such matrices. We denote by $P_{\mathbf{\Sigma}}$ the copula on $[0,1]^{2d}$ associated with the covariance matrix $\mathbf{\Sigma}$.

It is immediate that both P_{iid} and P_{CRN} are elements of $\mathscr{G}$. IID sampling corresponds to $\mathbf{\Sigma} = \mathbf{I}_{2d}$, or in other words $\mathbf{\Sigma}_{XY} = \mathbf{0}_d$, the $d \times d$ zero matrix. On the other hand, Condition 2 of §2.2 describing CRN sampling corresponds to $\mathbf{\Sigma}_{XY} = \mathbf{\Sigma}_{YX} = \mathbf{I}_d$.

We can consider X and Y to depend on the Gaussian random vector $(\mathbf{Z}_X, \mathbf{Z}_Y)$ directly. Thus, in order to simplify notation, we introduce functions $f, g : \mathbb{R}^{2d} \to \mathbb{R}$ given by

$$f(\mathbf{z}) = f_U(\Phi^{-1}(z[1]), \ldots, \Phi^{-1}(z[d])),$$
$$g(\mathbf{z}) = g_U(\Phi^{-1}(z[1]), \ldots, \Phi^{-1}(z[d])).$$

Examples 1 and 2, below, are simple cases where a Gaussian copula outperforms both CRN and independent sampling.

Example 1. Take $d = 2$, $f(\mathbf{Z}_X) = (Z_X[1] + Z_X[2])/\sqrt{2}$, and $g(\mathbf{Z}_Y) = Z_Y[1]$. Here X and Y have linear relationships to $\mathbf{Z}_X$ and $\mathbf{Z}_Y$, respectively. The random variable X depends on $\mathbf{Z}_X$ through both its components equally, whereas Y depends only upon the first component of $\mathbf{Z}_Y$. The admissible covariance matrix with $\mathbf{\Sigma}_{XY} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ defines an optimal, and in fact a perfect, Gaussian copula for this problem; this covariance matrix corresponds to setting $\mathbf{Z}_Y[1] = (Z_X[1] + Z_X[2])/\sqrt{2}$ and $\mathbf{Z}_Y[2] = (Z_X[1] - Z_X[2])/\sqrt{2}$, so that $X = Y$.

Observe that the columns of $\mathbf{\Sigma}_{XY}$ have L_2 norm 1. We may interpret this fact to mean that the optimal copula results in perfect correlation between the appropriate linear functions of $\mathbf{Z}_X$ and $\mathbf{Z}_Y$. This copula is strongly related to CRN sampling in that $\mathbf{U}_Y$ is a deterministic transformation of $\mathbf{U}_X$. We prove that this happens whenever f and g are linear in §2.6.

Example 2. Take $d = 1$, $f_U(U_X) = \chi_{[.5,.6]}(U_X)$, and $g_U(U_Y) = \chi_{[.6,.7]}(U_Y)$. Here, χ_A denotes the indicator function (characteristic function) of the set A. In order to maximize the covariance of these indicator random variables, we would like to have the events $U_X \in [.5, .6]$ and $U_Y \in [.6, .7]$ tend to occur at the same time. In fact, a copula satisfying $U_X \in [.5, .6] \Longleftrightarrow U_Y \in [.6, .7]$ would be perfect.

Unfortunately, there is no Gaussian copula satisfying this condition. The optimal Gaussian copula, in contrast, is given by the covariance matrix $\mathbf{\Sigma} = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$, where ρ is chosen to maximize the probability of $(Z_X, Z_Y) \in [\Phi^{-1}(.5), \Phi^{-1}(.6)] \times [\Phi^{-1}(.6), \Phi^{-1}(.7)]$ (equivalently, to maximize EXY). It is easy to see that $\rho = 1$ sets this probability to zero, so clearly $\rho = 1$ is not optimal. On the other hand, it is intuitive that $\rho > 0$ is desirable since we want Z_X and Z_Y to tend to have the same sign. Figure 2.1 shows that the optimal value of ρ is about .97. This value yields $\mathrm{Var}(X - Y) \approx .15$. Contrast this result with Example 1, where the optimal Gaussian copula had every column of $\mathbf{\Sigma}_{XY}$ having norm 1.

2.5 Finding an Optimal Gaussian Copula

The primary reason we have restricted attention to Gaussian copulae is that they are easily parameterized by a covariance matrix. This allows us to perform a numerical search for a locally optimal copula. We formulate the optimization problem in two distinct ways. Both involve maximizing the covariance between X and Y, but the underlying space over which the optimization is performed differs.

The first formulation is a nonlinear semidefinite program (NLP-SDP) (e.g., Kočvara and Stingl 2003). The decision variable in the optimization problem is the covariance matrix $\mathbf{\Sigma}$ of the Gaussian copula, which varies over the feasible region of admissible covariance matrices $\mathscr{S}_d$ as defined in (2.1). Let $\mathbf{Z}$ be a $2d$-dimensional standard multivariate normal random vector. We then wish to maximize $Eh(\mathbf{\Sigma}, \mathbf{Z})$,

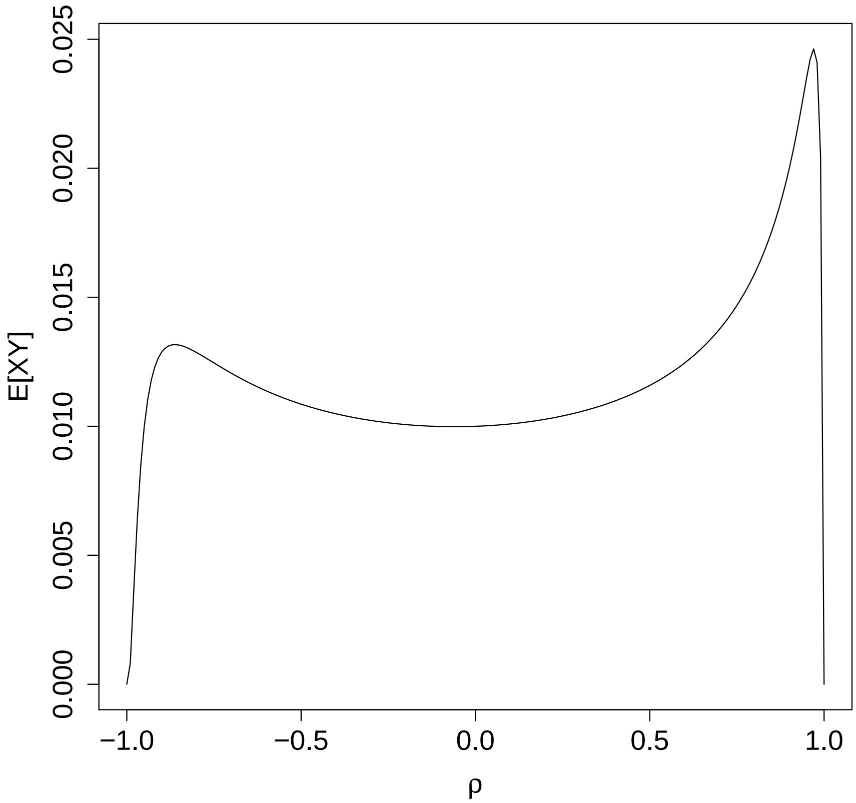


Figure 2.1: $E[XY]$ as a function of ρ.

where

$$h\left(\mathbf{\Sigma},\mathbf{Z}\right)= \\ f\left(\left(\mathbf{\Sigma}^{1/2}\mathbf{Z}\right)[1,\ldots,d]\right)g\left(\left(\mathbf{\Sigma}^{1/2}\mathbf{Z}\right)[d+1,\ldots,2d]\right). \quad (2.2)$$

Here, $\mathbf{\Sigma}^{1/2}$ denotes the Cholesky factor of $\mathbf{\Sigma}$.

One can view $\mathbf{\Sigma}^{1/2}$ as a differentiable function (admittedly complicated) of $\mathbf{\Sigma}$, so that (2.2) is differentiable in $\mathbf{\Sigma}$ for each fixed Z. One might then apply gradient-based methods for performing the optimization. Unfortunately, actually computing the gradient is difficult. One might resort to some rather complicated approach based on infinitesimal perturbation, or a more straightforward approach based on finite differences. In either case there are computational disadvantages, so we turn to a different formulation that seems more readily adapted to computation.

In our second formulation, rather than treating the covariance matrix $\mathbf{\Sigma}$ as the decision variable, we optimize over the space of all appropriate linear transformations of $\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{2d})$. The key to this formulation is the following proposition. A matrix M with at least as many rows as columns is called orthogonal if $M^T M$ gives the identity matrix.

prop 2.1. *Let $\mathbf{\Sigma}_{XY}$ and $\mathbf{M}_2$ be $d \times d$ matrices such that*

$$\mathbf{M} := \begin{bmatrix} \mathbf{\Sigma}_{XY} \\ \mathbf{M}_2 \end{bmatrix}$$

is orthogonal. Then the covariance matrix of

$$\begin{bmatrix} \mathbf{Z}_X \\ \mathbf{Z}_Y \end{bmatrix} := \begin{bmatrix} \mathbf{Z}[1, \ldots, d] \\ \mathbf{M}^T \mathbf{Z} \end{bmatrix}$$

is given by

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{I}_d & \mathbf{\Sigma}_{XY} \\ \mathbf{\Sigma}_{XY}^T & \mathbf{I}_d \end{bmatrix}.$$

Conversely, if $\mathbf{\Sigma}$ is admissible then there exists $\mathbf{M}_2$ such that

$$\begin{bmatrix} \mathbf{\Sigma}_{XY} \\ \mathbf{M}_2 \end{bmatrix}$$

is orthogonal.

Proof. For the first statement, we have

$$
\begin{aligned}
\operatorname{Cov}\begin{bmatrix}\mathbf{Z}_X & \mathbf{Z}_Y\end{bmatrix} &= E\begin{bmatrix}\mathbf{Z}_X\mathbf{Z}_X^T & \mathbf{Z}_X\mathbf{Z}^T\mathbf{M} \\ \mathbf{M}^T\mathbf{Z}\mathbf{Z}_X^T & \mathbf{M}^T\mathbf{Z}\mathbf{Z}^T\mathbf{M}\end{bmatrix} \\
&= \begin{bmatrix}\mathbf{I}_d & \begin{bmatrix}\mathbf{I}_d & \mathbf{0}_d\end{bmatrix}\mathbf{M} \\ \mathbf{M}^T\begin{bmatrix}\mathbf{I}_d \\ \mathbf{0}_d\end{bmatrix} & \mathbf{M}^T\mathbf{M}\end{bmatrix} \\
&= \mathbf{\Sigma}.
\end{aligned}
$$

Conversely, suppose $\begin{bmatrix}\mathbf{Z}_X & \mathbf{Z}_Y\end{bmatrix}$ has admissible covariance matrix $\mathbf{\Sigma}$. Then

$$
\begin{aligned}
\operatorname{Cov}\left(\mathbf{Z}_Y - \mathbf{\Sigma}_{XY}^T\mathbf{Z}_X\right) &= \begin{bmatrix}-\mathbf{\Sigma}_{XY}^T & \mathbf{I}_d\end{bmatrix}\mathbf{\Sigma}\begin{bmatrix}-\mathbf{\Sigma}_{XY} \\ \mathbf{I}_d\end{bmatrix} \\
&= \begin{bmatrix}-\mathbf{\Sigma}_{XY}^T & \mathbf{I}_d\end{bmatrix}\begin{bmatrix}\mathbf{0}_d \\ \mathbf{I}_d - \mathbf{\Sigma}_{XY}^T\mathbf{\Sigma}_{XY}\end{bmatrix} \\
&= \mathbf{I}_d - \mathbf{\Sigma}_{XY}^T\mathbf{\Sigma}_{XY},
\end{aligned}
$$

implying that the matrix on the right is positive semidefinite. Therefore, there exists $\mathbf{M}_2$ such that $\mathbf{M}_2^T\mathbf{M}_2 = \mathbf{I}_d - \mathbf{\Sigma}_{XY}^T\mathbf{\Sigma}_{XY}$ and hence $\mathbf{M}^T\mathbf{M} = \mathbf{\Sigma}_{XY}^T\mathbf{\Sigma}_{XY} + \mathbf{M}_2^T\mathbf{M}_2 = \mathbf{I}_d$. □

Proposition 2.1 demonstrates that we can compute an optimal Gaussian copula by solving an optimization problem on the space $\mathscr{V}_{2d,d}$ of $2d \times d$ orthogonal matrices rather than on $\mathscr{S}_d$. The space of such matrices is an example of a *Stiefel manifold*; see Edelman, Arias, and Smith (1999) for a discussion of the geometry of these and related manifolds, and of optimization algorithms thereon. In this formulation, our objective function (2.2) is replaced by

$$
h(\mathbf{M}, \mathbf{Z}) = f\left(\mathbf{Z}_X\right) g\left(\mathbf{M}^T\mathbf{Z}\right). \tag{2.3}
$$

Again we wish to maximize $Eh(\mathbf{M},\mathbf{Z})$, but this time the feasible region is $\mathbf{M} \in \mathscr{V}_{2d,d}$.

It is now straightforward to compute derivatives of $h(\cdot,\mathbf{Z})$ with respect to $\mathbf{M}$ for a fixed $\mathbf{Z}$. We have that

$$\begin{aligned} h'_{ij}(\mathbf{M},\mathbf{Z}) &:= \frac{\partial h(\mathbf{M},\mathbf{Z})}{\partial \mathbf{M}[i,j]} \\ &= f(\mathbf{Z}_X) g_j\left(\mathbf{M}^T\mathbf{Z}\right)\mathbf{Z}[i], \end{aligned}$$

where $g_j(x)$ is the partial derivative of $g(\cdot)$ with respect to the jth component, evaluated at x.

One can use these derivatives in various gradient-based optimization approaches such as stochastic approximation. Our optimization approach is based upon Sample Average Approximation (SAA) (e.g., Shapiro 2004). SAA is a general method for solving optimization problems of the form

$$\max_{x\in\mathcal{X}} Eh(x,\xi)$$

where ξ is a random object. Initially a small "pilot" sample $\xi_1,\ldots,\xi_m$ is generated. These values are then treated as fixed, and the optimization problem is replaced by

$$\max_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^{m} h(x,\xi_i).$$

Since the sample is fixed, the problem can be viewed as a deterministic optimization problem, and one can then employ specialized deterministic optimization algorithms to solve the problem. We use exactly this approach using optimization algorithms designed for differentiable functions over a Stiefel manifold. The solution to the optimization problem using a sample of size m, say, $\mathbf{Z}_1,\ldots,\mathbf{Z}_m$ yields a matrix $\mathbf{M}^*_m$ that defines a copula, which can then be used in a "production" run to actually compare the systems in question. Under mild regularity conditions, it is known that $\mathbf{M}_m^*$ will not only be a locally optimal solution for the sample-average problem, but will also be a nearly locally optimal solution for the true problem. See Shapiro (2004), Proposition 7, p. 363 and Bastin, Cirillo, and Toint (2006).

Example 3. We conclude this section with an example of a stochastic activity network (e.g., Avramidis and Wilson 1993). The network in Figure 2.2 is an abstraction of a set of jobs which must be completed. Each arc corresponds to a job. Nodes represent constraints on the order in which the jobs must be performed. All the jobs whose arcs enter a given node must be completed before any job whose arc leaves that node commences. The arcs are labelled by random variables corresponding to the length of time required by each task. Two nodes are distinguished as the source and the sink, respectively representing the state in which no tasks have begun and the state in which all tasks are completed. The total completion time for the set of all tasks is equal to the maximum length of all paths from source to sink.

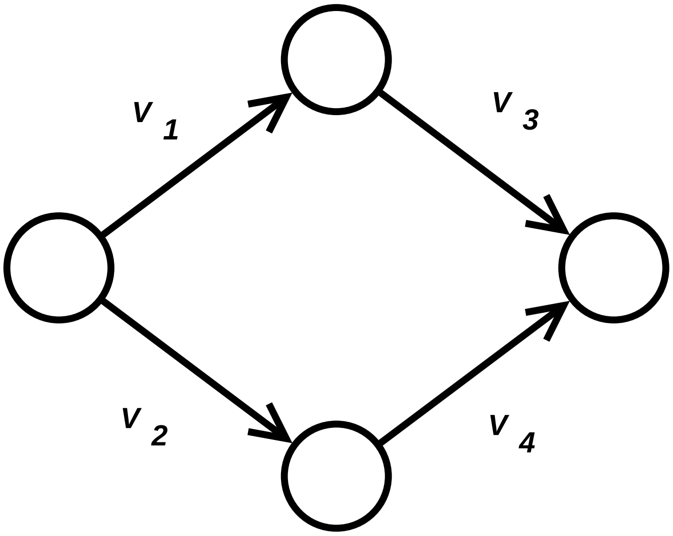


Figure 2.2: Stochastic Activity Network Example.

Let us compare two possible configurations of the stochastic activity network in Figure 2.2, where the different configurations correspond to different joint distributions on the activity times $V_1, \ldots, V_4$. In Configuration 1, the activity times are IID exponential with rate 1. In Configuration 2, V_1 and V_2 are as in Configuration 1, but V_3 and V_4 are exponential conditional on V_1 and V_2 with respective rates $\frac{1}{2}(1+V_2)$ and $\frac{1}{2}(1+V_1)$. Let X and Y respectively be the completion times of Configurations 1 and 2.

The functions connecting the underlying normal random variates to the service times are given by

$$\begin{aligned} f(\mathbf{z}) &= \max(-\log\Phi(z_1) - \log\Phi(z_3), \\ &\qquad -\log\Phi(z_2) - \log\Phi(z_4)) \\ g(\mathbf{z}) &= \max\big(-\log\Phi(z_1) - \frac{2}{1-\log\Phi(z_2)}\log\Phi(z_3), \\ &\qquad -\log\Phi(z_2) - \frac{2}{1-\log\Phi(z_1)}\log\Phi(z_4)\big). \end{aligned}$$

We solved this problem using the Stiefel manifold formulation with the freely available MATLAB procedure `sgmin` (Lippert and Edelman 1999), using the SAA framework sketched above. We solved it over both the spaces $\mathscr{V}_{2d,d}$ and $\mathscr{V}_{d,d}$ and achieved quite similar resulting covariance matrices. This strongly suggests that the optimal Gaussian copula is in fact a change of variables applied to CRN sampling.

The Gaussian copula returned by the optimization procedure is defined by (2.4). We performed longer runs, independent from the pilot, under IID sampling, CRN sampling, and the optimal Gaussian copula. The resulting variance of $(X-Y)$ in each case is given in Table 2.1.

$$\boldsymbol{\Sigma}_{XY} = \begin{bmatrix} .958 & -.038 & .160 & .237 \\ -.037 & .960 & .239 & .141 \\ -.158 & -.238 & .957 & -.048 \\ -.239 & -.143 & -.026 & .960 \end{bmatrix}. \tag{2.4}$$

Table 2.1: Example 3 results.

Sampling Strategy	Variance
IID	5.257
CRN	0.565
OPT	0.280

We can see that the optimal covariance matrix defined (2.4) returned by the optimization algorithm is quite close to that of CRN sampling. Although the random variables $Z_X[i]$ and $Z_Y[i]$, $i = 1, \dots, 4$, are not identical, they are very highly correlated. However, the difference in performance between the two copulae is great, with the optimal Gaussian copula resulting in more than a 50% reduction in variance.

2.6 Analysis of the Linear Case

The optimal copula given in Example 1 has the property that $\boldsymbol{\Sigma}_{XY}$ is itself an orthogonal matrix; equivalently, the lower d rows of this solution $\mathbf{M}$ are all zero. A natural question to ask is under what conditions we may assume that an optimal solution of this type exists. Knowledge of such conditions would allow the optimization problem to be solved on the smaller space $\mathscr{V}_{d,d}$. Moreover, the resulting copula can be sampled from using only d independent normal variates per sample, as opposed to the $2d$ normal variates required in the general case. We do not have a complete answer to this question at present, although we are able to show that a sufficient condition is for X and Y to be linear in $\mathbf{Z}_X$ and $\mathbf{Z}_Y$, respectively.

prop 2.2. *Let* $\mathbf{Z}$ *be a* $2d$*-dimensional random vector with mean zero and covariance matrix* $\mathbf{I}_{2d}$*. Suppose* f *and* g *are functions on* $\mathbb{R}^d$ *given by* $f(\mathbf{z}) = \mathbf{a}^T\mathbf{z}, g(\mathbf{z}) = \mathbf{b}^T\mathbf{z}$*, for some* $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$*. Then*

$$\max_{\mathbf{M}\in\mathscr{V}_{2d,d}} Ef(\begin{bmatrix}\mathbf{I}_d & \mathbf{0}_d\end{bmatrix}^T \mathbf{Z})g(\mathbf{M}^T\mathbf{Z})$$
$$= \max_{\mathbf{\Sigma}_{XY}\in\mathscr{V}_{d,d}} Ef(\begin{bmatrix}\mathbf{I}_d & \mathbf{0}_d\end{bmatrix}^T \mathbf{Z})g(\begin{bmatrix}\mathbf{\Sigma}_{XY} & \mathbf{0}_d\end{bmatrix}^T \mathbf{Z}).$$

Proof. Let LHS and RHS respectively denote the left- and right-hand sides of the desired equality. The inequality LHS $\geq$ RHS follows immediately from the fact that $\begin{bmatrix}\mathbf{\Sigma}_{XY} & \mathbf{0}_d\end{bmatrix} \in \mathscr{V}_{2d,d}$, so we need only prove the converse.

Let $\mathbf{M} \in \mathscr{V}_{2d,d}$ be arbitrary. Let us write

$$\begin{bmatrix}\mathbf{Z}_1 \\ \mathbf{Z}_2\end{bmatrix} \quad \text{and} \quad \begin{bmatrix}\mathbf{M}_1 \\ \mathbf{M}_2\end{bmatrix}$$

for $\mathbf{Z}$ and $\mathbf{M}$ respectively. Then

$$\begin{aligned} Ef(\begin{bmatrix}\mathbf{I}_d & \mathbf{0}_d\end{bmatrix}^T \mathbf{Z})g(\mathbf{M}^T\mathbf{Z}) &= E\mathbf{a}^T\mathbf{Z}_1\mathbf{b}^T\mathbf{M}^T\mathbf{Z} \\ &= E\mathbf{a}^T\mathbf{Z}_1\mathbf{Z}_1^T\mathbf{M}_1\mathbf{b} + E\mathbf{a}^T\mathbf{Z}_1\mathbf{Z}_2^T\mathbf{M}_2\mathbf{b} = \langle\mathbf{a}, \mathbf{M}_1\mathbf{b}\rangle. \end{aligned} \quad (2.5)$$

Now,

$$\begin{aligned} \langle\mathbf{a}, \mathbf{M}_1\mathbf{b}\rangle^2 &\leq \|\mathbf{a}\|^2\|\mathbf{M}_1\mathbf{b}\|^2 \\ &\leq \|\mathbf{a}\|^2\|\mathbf{M}_1\|_2^2\|\mathbf{b}\|^2 \\ &\leq \|\mathbf{a}\|^2\|\mathbf{b}\|^2. \end{aligned} \quad (2.6)$$

Here, $\|\mathbf{M}_1\|_2$ denotes the spectral norm (greatest absolute eigenvalue) of $\mathbf{M}_1$. The first inequality is Cauchy-Schwartz. The second inequality is a property of the spectral norm. The third inequality is proven as follows: if λ is an eigenvalue of $\mathbf{M}_1$ with corresponding eigenvector $\mathbf{w}$, then

$$\begin{aligned} 1 = \|\mathbf{w}\|^2 &= \mathbf{w}^T\mathbf{M}^T\mathbf{M}\mathbf{w} \\ &= \mathbf{w}^T(\mathbf{M}_1^T\mathbf{M}_1 + \mathbf{M}_2^T\mathbf{M}_2)\mathbf{w} \\ &= \lambda^2\|\mathbf{w}\|^2 + \mathbf{w}^T\mathbf{M}_2^T\mathbf{M}_2\mathbf{w} \\ &\geq \lambda^2, \end{aligned}$$

since $\mathbf{M}_2^T\mathbf{M}_2$ is positive semidefinite.

Combining (2.5) with (2.6) yields LHS $\leq \|\mathbf{a}\|\|\mathbf{b}\|$. Now let $v = \mathbf{b} - \frac{\|\mathbf{b}\|}{\|\mathbf{a}\|}\mathbf{a}$. Let $\mathbf{\Sigma}_{XY}$ be the Householder reflection induced by $\mathbf{v}$,

$$\mathbf{\Sigma}_{XY} = \mathbf{I}_d - \frac{2}{\|\mathbf{v}\|^2}\mathbf{v}\mathbf{v}^T.$$

It is easy to check that $\mathbf{\Sigma}_{XY} \in \mathscr{V}_{d,d}$ and that $\langle \mathbf{a}, \mathbf{\Sigma}_{XY}\mathbf{b}\rangle^2 = \|\mathbf{a}\|^2\|\mathbf{b}\|^2$. This implies $\|\mathbf{a}\|\|\mathbf{b}\| \leq$ RHS, completing the proof. $\square$

2.7 Conclusion

We have shown that it is possible to compute couplings of two random vectors that have IID components with the goal of minimizing the variance of the difference between real-valued functions of the random vectors. We use an underlying Gaussian copula because it is amenable to computation, although one could certainly consider other copula families as well. We have given simple examples where the gains beyond common random numbers are significant.

Chapter 3

Adaptive Control Variates for Multidimensional American Options

3.1 Introduction

Efficient pricing of American options remains a thorny issue in finance. This is true despite the fact that numerical techniques for solving this problem have been studied for decades – certainly at least since the binomial tree method of Cox et al. (1979). Both tree-based methods and PDE methods are very fast in low dimensions but do not extend well to higher-dimensional problems, arbitrary stochastic processes, or arbitrary payoff structures. In the last decade or so, attention has been turned to simulation techniques to solve such problems.

What makes American options much more difficult to price than their European counterparts, of course, is the embedded optimal stopping problem. Many of the early papers on using simulation to price American options therefore focus on this aspect of the computation. Carriere (1996) uses nonparametric regression techniques to approximate the value of continuing (i.e., not exercising) at every time step, proceeding backwards in time from expiry. This in turn produces a stopping rule: exercise only if the (known) value of exercise exceeds the (approximate) value of continuing. These ideas are developed further in Longstaff and Schwartz (2001) and Tsitsiklis and Van Roy (2001), both of which use linear regression on a fixed set of basis functions to approximate the continuation value.

The continuation value approximations obtained using these methods are not perfect, but they do yield feasible stopping policies. These policies therefore yield lower bounds on the true option price. Recently, Haugh and Kogan (2004), Rogers (2002), and Andersen and Broadie (2004) showed how to compute upper bounds on the option price via a martingale duality. Bolia and Juneja (2005) observed that this same martingale, if computed by function approximation instead of simulation-within-simulation, can serve as a simulation control variate and thereby provide variance reduction. This approach can be viewed as a special case of a class of martingale control variate methods introduced by Henderson and Glynn (2002).

The method introduced by Bolia and Juneja (2005) relies on finding a particular set of basis functions. To avoid internal simulations it is necessary that the basis functions be such that one can easily compute certain one-step conditional expectations. In related work, Rasmussen (2005) computes a control variate for the option price by using a carefully chosen European option (or several such options), evaluated at the exercise time of the American option being priced. Laprise et al. (2006) construct upper and lower piecewise linear approximations of the value function and compute the American option price using a sequence of portfolios of European options. Their method only works in one dimension, though. An earlier use of European options as control variates for American options appears in Broadie and Glasserman (2004), wherein the European options in question expire in a single time step and employed at each step of a stochastic mesh scheme.

The work we present here, like that of Bolia and Juneja (2005), can be thought of as a "primal-dual" method, in the sense of Andersen and Broadie (2004). The martingale-based control variate is used both to improve the quality of the lower bound *and* to derive the upper bound. In our work, as well as that of Andersen and Broadie (2004), the upper bound solution is derived by first considering a suboptimal stopping strategy, and then deriving a corresponding martingale. Thus, a poor choice of stopping strategy will never be "rescued" by the fact that an upper bound is available. However, unlike Andersen and Broadie (2004), our upper bound solutions do not involve any additional simulation trials. As a result, the quality of the upper bound depends not only on the quality of the suboptimal stopping times but *also on a functional approximation* for the martingale from which the upper bound arises. In that sense, our work can also be thought of as primarily a variance reduction technique for lower bound methods, albeit one which produces an upper bound for free.

Our contribution is to identify a technique for computing the control variate that possesses the desired tractability property in a quite general setting. Moreover, construction of the control variate is more or less automatic; once it has been done for one pricing problem it can be extended to other problems without much effort. We demonstrate these extensions in detail for various basket options, barrier options, and Asian options in both a Black-Scholes and stochastic volatility (Heston 1993) model.

Rogers (2002) commented that the selection of the dual martingale may be "more art than science." We contend that our approach takes a bit of the art out of this process and injects, if not science, at least some degree of automation to the procedure.

The remainder of this paper is organized as follows. Section 3.2 gives some mathematical preliminaries, recalls the pricing algorithm of Longstaff and Schwartz (2001), defines the martingales that we work with and clarifies their linkage with the pricing problem. Section 3.3 discusses multivariate adaptive regression splines (Friedman 1991), or MARS, and discriminant analysis, which are the techniques we adapt to construct martingales. Section 3.4 describes the algorithm in detail. Section 3.5 gives a number of examples, and we offer some conclusions in Section 3.6.

3.2 Mathematical Framework

As in most papers that discuss simulation applied to American option pricing, we actually consider the problem of pricing a Bermudan option, which differs from its American counterpart in that it may be exercised only at a finite set of points in time. To simplify notation, we assume that these times are the evenly spaced steps $t = 0, \ldots, T$.

Let $(X_t : t = 0, \ldots, T)$ be an $\mathbb{R}^d$-valued process on a filtered probability space $(\Omega, \mathscr{F}, \mathbb{P})$, where $\mathscr{F} = (\mathscr{F}_t : t = 0, \ldots, T)$ is the natural filtration of (X_t). We assume (X_t) to be Markov, enlarging the state space if necessary to ensure this. We treat X_0 as deterministic, so $\mathscr{F}_0$ is taken to be trivial. Let r be the riskless interest rate which we assume to be constant and, to simplify notation, normalized so that if $s < t$, the time-s dollar value of \$1 to be delivered at time t is $e^{-r(t-s)}$. We assume that the market is arbitrage-free and work exclusively with a risk-neutral (pricing) measure $\mathbb{Q}$ with the same null sets as $\mathbb{P}$. See e.g., Duffie (2001) or Glasserman (2004) for details on risk-neutral pricing.

Let the known function $g : \{0, \ldots, T\} \times \mathbb{R}^d$ satisfy $g(t, \cdot) \geq 0$ and $Eg^2(t, X_t) < \infty$ for all $t = 0, \ldots, T$. We interpret $g(t, X_t)$ to be the value of exercising the option at time t in state X_t. Let $\mathcal{T}(t)$ be the set of all $\mathscr{F}$-stopping times valued in

$\{t, \ldots, T\}$. Then the Bermudan option pricing problem is to compute Q_0, where

$$Q_t = \sup_{\tau \in \mathcal{T}(t)} E_t \left[e^{-r(\tau - t)} g\left(\tau, X_\tau\right) \right],$$

for $t = 0, \ldots, T$. We recall some theory about American and Bermudan options; again see e.g., Duffie (2001) for details. The above optimal stopping problem admits a solution $\tau_t^* \in \mathcal{T}(t)$, so that

$$Q_t = E_t \left[e^{-r(\tau_t^* - t)} g\left(\tau_t^*, X_{\tau_t^*}\right) \right]$$

for each $t = 0, \ldots, T$. Moreover, the Q_t's satisfy the backward recursion

$$\begin{aligned} Q_T &= g(T, X_T), \\ Q_t &= \max\left\{ g(t, X_t), e^{-r} E_t Q_{t+1} \right\}, \end{aligned}$$

for $t = 0, \ldots, T-1$. Therefore, the optimal stopping times $\tau_0^*, \ldots, \tau_T^*$ satisfy

$$\begin{aligned} &\tau_T^* \equiv T, \\ &\tau_t^* = \begin{cases} t & \text{if } g(t, X_t) \geq e^{-r} E_t Q_{t+1}, \\ \tau_{t+1}^* & \text{otherwise,} \end{cases} \end{aligned}$$

for $t = 0, \ldots, T-1$. An easy consequence of this is

$$s < t \leq \tau_s^* \Longrightarrow \tau_s^* = \tau_{s+1}^* = \cdots = \tau_t^*. \tag{3.1}$$

3.2.1 The Longstaff-Schwartz Method

The least-squares Monte Carlo (LSM) method of Longstaff and Schwartz (2001) provides an approximation to the optimal stopping times (τ_t^*) and hence to the option price process (Q_t). Since the resulting stopping times (τ_t) are suboptimal for the original problem, the value of following such a stopping strategy provides a lower bound on the true price process. Following the notation of Andersen and Broadie (2004), we denote the lower bound process by

$$L_t = E_t e^{-r(\tau_t - t)} g\left(\tau_t, X_{\tau_t}\right).$$

We now recall the procedure by which LSM computes the stopping times (τ_t) and hence (L_t). Let $\phi_0, \phi_1, \ldots,$ be a collection of functions from $\mathbb{R}^d$ to $\mathbb{R}$ such that $\phi_0 \equiv 1$ and $\{\phi_i(X_t) : i = 0, 1, \ldots\}$ form a basis for $L^2(\Omega, \sigma(X_t), \mathbb{Q})$ for all $t = 1, \ldots, T$. The algorithm proceeds as follows. Denote $\boldsymbol{\phi} = (\phi_0, \ldots, \phi_k)$, for some fixed k. Generate a set of N paths $\{X_t(n) : t = 0, \ldots, T; n = 1, \ldots, N\}$. Set $\tau_T(n) = T$ and $L_T(n) = g(T, X_T(n))$ for $n = 1, \ldots, N$. Then recursively estimate

$$\boldsymbol{\alpha}_t = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \sum_{n=1}^{N} 1_{[g(t, X_t(n)) > 0]} \left(\boldsymbol{\alpha}' \boldsymbol{\phi}(X_t(n)) - L_{t+1}(n)\right)^2, \tag{3.2}$$

$$\tau_t(n) = \begin{cases} t & \text{if } g(t, X_t(n)) > \left[\boldsymbol{\alpha}_t' \boldsymbol{\phi}(X_t(n))\right]_+ \\ \tau_{t+1}(n) & \text{otherwise,} \end{cases}$$

$$L_t(n) = \begin{cases} g(t, X_t(n)) & \text{if } \tau_t(n) = t, \\ e^{-r} L_{t+1}(n) & \text{otherwise,} \end{cases}$$

for $t = T-1, \ldots, 0$. The regression in (3.2) is performed only on those paths which have positive exercise value at time t, thus (we hope) producing a better fit on the paths that actually matter than we would obtain if we performed the regression on the complete set of paths. We shall comment on this point in Section 3.4.1 when we describe our version of the algorithm with the control variate.

The idea behind the LSM algorithm is that if τ_t is close to the true optimal stopping time τ_t^*, then the lower-bounding value process L_t is close to Q_t. It is shown in Clément et al. (2002) both that the approximations τ_t converge to τ_t^*

and that the approximations L_t converge to Q_t as the number of basis functions used $k \to \infty$.

3.2.2 Martingales and Variance Reduction

As we have already noted, the stopping times (τ_t) obtained in the LSM method are suboptimal, and so the option prices (L_t) implied by the algorithm are lower bounds on the true option prices (Q_t). To obtain an upper bound we employ a martingale duality result developed independently by Haugh and Kogan (2004) and Rogers (2002). Let $\pi = (\pi_t : t = 0, \ldots, T)$ denote a martingale with respect to $\mathscr{F}$. By the optional sampling theorem, for any $t \geq 0$,

$$\begin{aligned} Q_t &= e^{rt} \sup_{\tau \in \mathcal{T}(t)} E_t \left[e^{-r\tau} g(\tau, X_\tau) - \pi_\tau + \pi_\tau \right] \\ &= e^{rt} \sup_{\tau \in \mathcal{T}(t)} E_t \left[e^{-r\tau} g(\tau, X_\tau) - \pi_\tau \right] + e^{rt} \pi_t \\ &\leq e^{rt} E_t \max_{s=t,\ldots,T} \left[e^{-rs} g(s, X_s) - \pi_s \right] + e^{rt} \pi_t \\ &=: U_t. \end{aligned} \tag{3.3}$$

The martingale π here is arbitrary, and any such choice yields an upper bound. We next give a class of martingales from which to choose.

Let $h_t : \mathbb{R}^d \to \mathbb{R}$ be such that $E|h_t(X_t)| < \infty$ for each $t = 0, 1, \ldots, d$. Define $\pi_0 = 0$, and for $t = 1, \ldots, T$, set

$$\pi_t = \sum_{s=1}^{t} e^{-rs} (h_s(X_s) - E_{s-1} h_s(X_s)). \tag{3.4}$$

Evidently (π_t) is a martingale, and can be used to obtain an upper bound on the option price as in (3.3).

Such martingales can also be used to great effect as control variates in estimating the lower bound process. Recall that $L_t = E_t e^{-r(\tau_t - t)} g(\tau_t, X_{\tau_t})$ for each t, and so conditional on $\mathscr{F}_t$, we can compute L_t by averaging conditionally independent replicates of $e^{-r(\tau_t - t)} g(\tau_t, X_{\tau_t})$. Proposition 3.1 shows that if we choose the function h_t so that $h_t(X_t) = L_t$ for each t, then the martingale difference $\pi_{\tau_t} - \pi_t$ is a perfect control variate, in the sense that it is perfectly correlated with $e^{-r(\tau_t - t)} g(\tau_t, X_{\tau_t})$, conditional on $\mathscr{F}_t$. This generalizes a comment in Bolia and Juneja (2005), who show that the proposition holds in the case $t = 0$.

prop 3.1. *Suppose that h_t is chosen so that $h_t(X_t) = L_t$ for each $t = 1, \ldots, T$. Then the martingale $\pi = (\pi_t : t = 0, \ldots, T)$ defined in (3.4) satisfies*

$$\pi_{\tau_t} - \pi_t = e^{-r\tau_t} g(\tau_t, X_{\tau_t}) - e^{-rt} L_t$$

for each $t = 0, 1, \ldots, T$.

Proof. First observe that on the event $[\tau_t = t]$, both sides of the equality we are trying to prove are zero. Hence, it suffices to prove that the result holds in the continuation region, i.e.,

$$1_{[\tau_t > t]} \left(\pi_{\tau_t} - \pi_t\right) = 1_{[\tau_t > t]} \left(e^{-r\tau_t} g\left(\tau_t, X_{\tau_t}\right) - e^{-rt} L_t\right). \tag{3.5}$$

Now, if $s \in \{t + 1, \ldots, T\}$, then

$$\begin{aligned}
1_{[\tau_t \geq s]} E_{s-1} L_s &= 1_{[\tau_t \geq s]} E_{s-1} \left[E_s e^{-r(\tau_s - s)} g\left(\tau_s, X_{\tau_s}\right)\right] \\
&= E_{s-1} 1_{[\tau_t \geq s]} e^{-r(\tau_s - s)} g\left(\tau_s, X_{\tau_s}\right) \\
&= e^r E_{s-1} 1_{[\tau_t \geq s]} e^{-r(\tau_{s-1} - (s-1))} g\left(\tau_{s-1}, X_{\tau_{s-1}}\right) \\
&= e^r 1_{[\tau_t \geq s]} L_{s-1},
\end{aligned}$$

where the penultimate equality uses the fact that

$$s < t \leq \tau_s \Longrightarrow \tau_s = \tau_{s+1} = \cdots = \tau_t,$$

analogous to (3.1). Therefore,

$$\begin{aligned}
\pi_{\tau_t} - \pi_t &= \sum_{s=t+1}^{T} 1_{[\tau_t \geq s]} e^{-rs} \left(L_s - E_{s-1} L_s\right) \\
&= \sum_{s=t+1}^{T} 1_{[\tau_t \geq s]} e^{-rs} \left(L_s - e^{r} L_{s-1}\right) \\
&= \sum_{s=t+1}^{T} 1_{[\tau_t \geq s]} e^{-rs} L_s - \sum_{s=t}^{T-1} 1_{[\tau_t \geq s+1]} e^{-rs} L_s \\
&= 1_{[\tau_t = T]} e^{-rT} L_T + \sum_{s=t+1}^{T-1} e^{-rs} L_s \left(1_{[\tau_t \geq s]} - 1_{[\tau_t \geq s+1]}\right) - 1_{[\tau_t > t]} e^{-rt} L_t \\
&= \sum_{s=t+1}^{T} e^{-rs} L_s 1_{[\tau_t = s]} - 1_{[\tau_t > t]} e^{-rt} L_t .
\end{aligned}$$

So

$$\begin{aligned}
1_{[\tau_t > t]} \left(\pi_{\tau_t} - \pi_t\right) &= 1_{[\tau_t > t]} \sum_{s=t+1}^{T} 1_{[s = \tau_t]} e^{-rs} L_s - 1_{[\tau_t > t]} e^{-rt} L_t \\
&= 1_{[\tau_t > t]} \left(e^{-r\tau_s} L_{\tau_s} - e^{-rt} L_t\right),
\end{aligned}$$

proving (3.5). □

Proposition 3.1 shows that conditional on $\mathscr{F}_t$, we can estimate L_t with zero (conditional) variance by

$$e^{-r(\tau_t - t)} g(\tau_t, X_{\tau_t}) - e^{rt}(\pi_{\tau_t} - \pi_t).$$

Since $\mathscr{F}_0$ is the trivial sigma field, by taking $t = 0$ we get a zero variance estimator of L_0, the lower bound on the option price at time 0. In other words, $e^{rt}\left(\pi_{\tau_t} - \pi_t\right)$ is the "perfect" additive control variate for estimating L_t from $e^{-r(\tau_t - t)} g(\tau_t, X_{\tau_t})$.

Of course, we cannot set $h_t(X_t) \equiv L_t$, since we are trying to compute L_t in the first place. But this observation motivates us to search for a set of functions $\{\hat{h}_t\}$ such that

$$\hat{h}_t(X_t) \approx L_t$$

for each t. (In this paper the approximation is in the mean-square sense.) Let us write $\hat{L}_t$ for $\hat{h}_t(X_t)$, and let the induced martingale be $\hat{\pi} = (\hat{\pi}_t : t = 0, 1, \ldots, T)$, where $\hat{\pi}_0 = 0$ and

$$\hat{\pi}_t = \sum_{s=1}^{t} e^{-rs} \left(\hat{L}_s(X_s) - E_{s-1} \hat{L}_s(X_s) \right).$$

We use the approximately optimal martingale $\hat{\pi}$ evaluated at time τ_0 as a control variate in estimating L_0, as indicated by the remark following Proposition 3.1; details are given in Section 3.4.

Observe that there are two distinct approximations being performed. The one described in the preceding paragraph approximates the value of the option at time t by a (more tractable) function of the underlying state X_t. In contrast, (3.2) projects the realized value of the option at time $t+1$ onto a certain space of random variables measurable with respect to $\mathcal{F}_t$. In the language of Glasserman and Yu (2004), the approximation used to compute the martingale is "regression later," whereas the approximation (3.2) used for the stopping strategy is "regression now."

In addition to serving as a control variate, the martingale $\hat{\pi}$ begets an upper bound on the true option price, as in (3.3). Andersen and Broadie (2004) show that the martingale π is the optimal one to use in computing the upper bound, and indeed that the inequality in (3.3) would actually be an equality if we had $\tau_t = \tau_t^*$ almost surely, for $t = 0, \ldots, T$. This motivates the use of the martingale $\hat{\pi}$ to compute the upper bound. We note that the same observation is made in Bolia and Juneja (2005).

To compute the martingale $\hat{\pi}$, we need to be able to compute the conditional expectation $E_{s-1} h_s(X_s)$ efficiently. We restrict the class of functions $\{h_t\}$ considered so that these conditional expectations can be evaluated without the need to resort to further simulation, in the same spirit as Bolia and Juneja (2005) and

Rasmussen (2005). Bolia and Juneja (2005) use a particular parametric form for h_t which is easily fit by least squares, but is tightly coupled with the specific stochastic process considered. Rasmussen (2005) chooses h_t to be the value of a European option, or a combination of several European options, that are highly correlated with the American option being priced. Indeed, in many examples Rasmussen (2005) simply chooses h to be given by $h_t(X_t) = E_t g(T, X_T)$ so that $\pi_t = e^{-rt} E_t g(T, X_T) - E g(T, X_T)$. The success of their method, therefore, depends on the ability to find particular European options which can be easily priced and which correlate well with the American option in question. Our method also involves the pricing of European options in a sense, although not necessarily options on traded assets. Like Broadie and Glasserman (2004), the European options we use as control variates each expire after a single time step. These options are automatically selected using the MARS fitting procedure, and in general are priced easily. We now explore MARS.

3.3 MARS and Extensions

Multivariate adaptive regression splines (Friedman 1991), or MARS, is a nonparametric regression technique that has enjoyed widespread use in a variety of applications since its introduction. For example, Chen et al. (1999) use MARS to approximate value functions of a stochastic dynamic programming problem, although for a different purpose than we do here.

Given observed responses $y(1), \ldots, y(N) \in \mathbb{R}$ and predictors $x(1), \ldots, x(N) \in \mathbb{R}^d$, MARS fits a model of the form

$$y \approx \hat{f}(x) = \alpha_0 + \sum_{m=1}^{M_1} \alpha_{1,m} f_{1,m}(x) + \sum_{m=1}^{M_2} \alpha_{2,m} f_{2,m}(x) + \cdots + \sum_{m=1}^{M_p} \alpha_{p,m} f_{p,m}(x).$$

Each function $f_{1,m}$ takes one of two forms,

$$f_{1,m}(x) \in \left\{ \left(\left[x^{(i)} - x^{(i)}(n) \right] \right)_+ , \left(\left[x^{(i)}(n) - x^{(i)} \right] \right)_+ \right\},$$

for some $i = 1, \ldots, d$, and some $n = 1, \ldots, N$. Here, $x^{(i)}$ denotes the i'th coordinate of x. Each function $f_{j,m}$ for $j > 1$ is a product of functions used in previous sums so that the total degree is j. In our setting, we take $p = 1$ so the fitted model can be written

$$y \approx \hat{f}(x) = \alpha_0 + \sum_{i=1}^{d} \sum_{j=1}^{J_i} \alpha_{i,j} \left(q_{i,j} \left[x^{(i)} - k_{i,j} \right] \right)_+ , \tag{3.6}$$

where $q_{i,j} \in \{-1, +1\}$ and the knots $k_{i,j}$ are chosen from the data: $k_{i,j} \in \{x^{(i)}(n) : n = 1, \ldots, N\}$, for each $i = 1, \ldots, d$ and each $j = 1, \ldots, J_i$. A function with the form (3.6) may be called an additive linear spline.

We present a simplified version of the MARS fitting algorithm here, as we are only concerned with the $p = 1$ case. Full details are given in Friedman (1991), and a summary can be found in Hastie et al. (2001). MARS produces a fitted model by proceeding in a stepwise manner. At each step, the algorithm attempts to add each possible pair of basis functions[1]

$$\left\{ \left(x^{(i)} - x^{(i)}(n) \right)_+ , \left(x^{(i)}(n) - x^{(i)} \right)_+ \right\}$$

in turn for $n = 1, \ldots, N$ and $i = 1, \ldots, d$. It adds a basis function if the improvement in fit from adding that function exceeds a given threshold, up to a specified number of basis functions M_{max}. Upon completion of this procedure, the algorithm prunes some of the basis functions it has selected if doing so will improve the weighted mean-square error criterion

$$\frac{\frac{1}{N} \sum_{n=1}^{N} \left(y_n - \hat{f}(x_n) \right)^2}{(1 - \frac{CM_1+1}{N})^2},$$

[1] In fact, the algorithm sorts the $x_n(i)$'s and skips a small number of observations between each knot it considers. This helps to prevent over-fitting and offers some computational benefits as well.

where C is a specified penalty parameter.

Friedman (1991) argues that the computation time of the fitting algorithm has an upper bound proportional to $dNM_{\max}^4$. Our implementation of MARS takes $M_{\max} = 21 \vee (2d+1)$, so for $d < 10$ the computational time is simply proportional to dN; for higher dimensions, it is proportional to $d^5 N$. However, in our experiments, we have found that the threshold criterion is often met before $M_{\max}$ basis functions are even considered, so even though the upper bounds discussed above are valid, they may be quite pessimistic.

3.3.1 Computing the Approximating Martingale

Suppose we have used MARS to fit

$$\hat{h}_t(x) = \alpha_0 + \sum_{i=1}^{d} \sum_{j=1}^{J_i} \alpha_{i,j} \left(q_{i,j} \left[x^{(i)} - k_{i,j} \right] \right)_+$$

for each time step $t = T, \ldots, 1$. Then for each $t = 1, \ldots, T$, the t'th increment of the resulting martingale $(\hat{\pi}_t)$ is given by

$$\hat{\pi}_t - \hat{\pi}_{t-1} = e^{-rt} \sum_{i=1}^{d} \sum_{j=1}^{J_i} \alpha_{i,j} \left(\left(q_{i,j} \left[X_t^{(i)} - k_{i,j} \right] \right)_+ - E_{t-1} \left[\left(q_{i,j} \left[X_t^{(i)} - k_{i,j} \right] \right)_+ \right] \right), \tag{3.7}$$

where we have suppressed the dependence of the fitted parameters on the time step t in the notation. Having simulated, say, $X_s(1), \ldots, X_s(N')$, $s = 1, \ldots, T$, it is evident how to compute the first term inside the sum in (3.7). The second term can be computed explicitly as long as we can compute expressions of the form

$$E_{t-1} \left[\left(X_t^{(i)} - k \right)_+ \right]. \tag{3.8}$$

But this is nothing but the expected value of a vanilla European call option on a *single* underlying random variable. Such conditional expectations can often be computed very easily. Even if the underlying random variables have complex dynamics, such as arises in a stochastic volatility model, we may be able to simplify the problem enough by selecting our discretization scheme carefully so that an answer is within reach. Typically, this will involve replacing the state variable X_t with some transformation of $\log X_t$. See Section 3.5 for specific examples of how we compute the conditional expectation.

3.3.2 An Extension of MARS

The function approximation (3.6) is separable in $\{x^{(i)} : i = 1, \ldots, d\}$. Of course, the function $L_t = L_t(X_t)$ we are trying to approximate will not be separable in general. Indeed, even if the payoff function g is separable, we cannot expect that $L_t(X_t)$ will be separable except for the case $t = T$. For example, consider the case $t = T - 1$. Here,

$$Q_{T-1} = Q_{T-1}(X_{T-1}) = \max\left\{g\left(T-1, X_{T-1}\right), e^{-r}E_{T-1}\left[g(T, X_T)\right]\right\}.$$

So even if g is separable, and even if $E_{T-1}g(T, X_T)$ is separable (which it may not be if there is dependence in the components of X_T), Q_{T-1} will typically not be separable, as the maximum of two separable functions need not be separable.

Intuitively, separability of g is equivalent to the European version of the option being decomposable into options on the individual components of X. Separability of Q_t for $t < T$, on the other hand, would mean that *the decision of whether to exercise early could be made separately for these options*, which is not the case. Since L_t can be made arbitrarily close to Q_t by employing sufficiently many basis functions, it follows that L_t will not be separable either. Thus, the best we can ever hope for with the approximation (3.6) is to obtain an approximation to the projection of $L_t = L_t(\cdot)$ on the space of separable functions. In particular, $E\left(\hat{L}_t - L_t\right)^2$ may be large no matter how much effort is spent on computing $\hat{L}_t$. This fact suggests that MARS may produce inadequate approximations to the optimal martingale.

In order to at least partially address this issue, we consider a more general form of the approximating multivariate linear spline,

$$y \approx \hat{f}(x) = \alpha_0 + \sum_{j=1}^{J} \alpha_j \left[a_j' x - k_j\right]_+ , \tag{3.9}$$

where we have additional parameters $a_j \in \mathbb{R}^d$, $j = 1, \ldots, d$, to estimate. This is quite similar to the form (3.6), except that now we consider linear combinations of the x's as predictors. One can think of the a_j vectors as giving a reparameterization of the state variables. If we a priori choose the a_j's, then the problem essentially reduces to the previous one. But this would require user intervention. We prefer an automated procedure, although one can certainly reparameterize manually before invoking our approach.

The following proposition indicates that it is possible to achieve good function approximations with expressions of the form (3.9).

prop 3.2. *Suppose X is an $\mathbb{R}^d$-valued random variable, and $f : \mathbb{R}^d \to \mathbb{R}$ satisfies $Ef^2(X) < \infty$. Then for any $\epsilon > 0$ there is a function $\hat{f}$ of the form* (3.9) *such that $E(f(X) - \hat{f}(X))^2 < \epsilon$.*

Proof. Jones (1987) shows that there exists a sequence of vectors

$$\left(a_m \in \mathbb{R}^d : m = 1, 2, \ldots\right)$$

such that

$$E\left(f(X) - \sum_{j=1}^{m} g_j\left(a_j' X\right)\right)^2 \longrightarrow 0$$

as $m \to \infty$. Here, the functions $(g_m : \mathbb{R} \to \mathbb{R} : m = 1, 2, \ldots,)$ are given recursively by

$$g_m(z) = E\left[f(X) - \sum_{j=1}^{m-1} g_j(a_j'X) \;\middle|\; a_m'X = z \right].$$

Accordingly, choose m sufficiently great such that $E\left(f(X) - \sum_{j=1}^{m} g_j(a_j'X)\right)^2 < \epsilon/2$. By induction, $Eg_j^2(a_j'X) < \infty$ for $j = 1, \ldots, m$. Since continuous functions are dense in L^2 (Rudin 1987, Theorem 3.14), we conclude from the Stone-Weierstrass Theorem that there exist linear splines $\hat{g}_j : \mathbb{R} \to \mathbb{R}$ such that

$$E\left(g_j(a_j'X) - \hat{g}_j(a_j'X)\right)^2 < \epsilon/2^{j+1},$$

for each $j = 1, \ldots, m$. The result now follows from the triangle inequality and the observation that, in one dimension, any linear spline can be written in the form (3.9). □

The expression (3.9) can be thought of as a specific example of a projection pursuit regression fit (Friedman and Stuetzle 1981; Friedman et al. 1983) using truncated linear splines as its univariate basis functions. Projection pursuit regression methods typically estimate the linear directions $a_1, \ldots, a_J$ and the remaining parameters simultaneously. This requires numerical optimization and can be slow. We instead adopt the simpler approach of Zhang et al. (2003), who first identify candidate a_j's and then run the MARS fitting algorithm with $x = (x(1), \ldots, x(d))$ replaced by $(a_1'x, \ldots, a_J'x)$. Zhang et al. (2003) provide two methods for selecting the a_r's. We consider the method that uses linear discriminant analysis, or LDA (Fisher 1936). Given responses $y(1), \ldots, y(N) \in \mathbb{R}$ and predictors $x(1), \ldots, x(N) \in \mathbb{R}^d$, choose some $\tilde{n} \in \{1, \ldots, N\}$ and define the corresponding LDA direction to be

$$S_x^{-1}\left[\frac{1}{|\{n : y(n) < y(\tilde{n})\}|} \sum_{y(n) < y(\tilde{n})} x(n) - \frac{1}{N - |\{n : y(n) < y(\tilde{n})\}|} \sum_{y(n) \geq y(\tilde{n})} x(n)\right],$$

where S_x denotes the sample covariance matrix of the $x(n)$'s. Observe that the bracketed term is nothing but the vector connecting the centroids of the two subpopulations of predictors.

This idea can be extended by performing LDA on the second moments of the predictor variables, leading to directions given by the eigenvectors of

$$S_x^{-1} \left[S_{x(n)|[y(n)<y(\tilde{n})]} - S_{x|[y(n)\geq y(\tilde{n})]} \right] S_x^{-1}, \tag{3.10}$$

where $S_{x|A}$ is the conditional sample covariance matrix of x given A. Zhang et al. (2003) argue that one typically only needs the eigenvectors of (3.10) corresponding to the two or three greatest magnitude eigenvalues.

We have found that including linear combinations of the components of X_t when estimating the approximation $\hat{L}_t$ as described in this section provides a dramatic improvement over the "vanilla" MARS fit in terms of reducing variance. This is most notable in the case of basket options (see Section 3.5.2), where the value functions are highly non-separable.

3.4 The Algorithm

We now describe how these pieces are put together to compute lower and upper bounds for the Bermudan option price. Like Bolia and Juneja (2005), we use a two phase procedure. In phase one, we compute the suboptimal stopping times $\tau_0, \ldots, \tau_{T-1}$ and the approximate value functions $\hat{L}_1, \ldots, \hat{L}_T$, working backwards from time T. This is done with a small number of simulation trials. In phase two, we run a large number of simulation trials to estimate the lower bound of the option price,

$$E\left[e^{-r\tau_0} g\left(\tau_0, X_{\tau_0}\right) - \hat{\pi}_{\tau_0}\right],$$

and the upper bound,

$$E \max_{t=0,\ldots,T} \left[e^{-rt} g(t, X_t) - \hat{\pi}_t\right].$$

Before presenting the algorithm, we point out how the control variate can be used not only to estimate L_0 but also to improve our estimates of the stopping times (τ_t).

3.4.1 Using the Control Variate to Estimate the Stopping Times

Since for each $t = 1, \ldots, T$, the random variable $\hat{\pi}_t$ can be computed without knowing $\tau_0, \ldots, \tau_{t-1}$, we can in fact use a modified version of the control variate for computing stopping times in phase one as well as for estimating the bounds on the price in phase two. (A similar idea is explored in Rasmussen 2005.) Specifically, we replace the approximation (3.2) by

$$\boldsymbol{\alpha}_t = \operatorname*{argmin}_{\boldsymbol{\alpha}} \sum_{n=1}^{N} 1_{[g(t,X_t(n))>0]} \left(\boldsymbol{\alpha}' \boldsymbol{\phi}(X_t(n)) - \left[L_{t+1}(n) - e^{r(t+1)}(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t)\right]\right)^2. \tag{3.11}$$

For the purposes of this regression, our estimate of L_{t+1} comes from samples of $e^{-r(\tau_{t+1}-(t+1))} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right)$, so we can write the predictors in the regression (3.11) as

$$Z_t\left(e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right),$$

where $Z_t := 1_{[g(t,X_t)>0]} e^{r(t+1)}$.

We now provide evidence that this actually improves (or at least, does not worsen) our stopping time estimates. For the time being, let us ignore the term Z_t. Evidently,

$$E_t e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) = E_t\left(e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right), \tag{3.12}$$

since $\hat{\pi}$ is a martingale. Moreover, by Proposition 3.1,

$$\pi_{\tau_{t+1}} - \pi_t = \left(\pi_{\tau_{t+1}} - \pi_{t+1}\right) + \left(\pi_{t+1} - \pi_t\right) = e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) - e^{-r(t+1)} E_t L_{t+1},$$

so

$$\operatorname{Var}_t\left[e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) - \left(\pi_{\tau_{t+1}} - \pi_t\right)\right] = \operatorname{Var}_t\left[E_t L_{t+1}\right] = 0,$$

where $\operatorname{Var}_t[\cdot]$ denotes the conditional variance $E_t(\cdot)^2 - E_t^2(\cdot)$. Therefore,

$$\operatorname{Var}_t\left[e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right] = \operatorname{Var}_t\left[\left(\pi_{\tau_{t+1}} - \pi_t\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right].$$

Taking expectations and invoking the variance decomposition formula gives

$$\begin{aligned}
&E \operatorname{Var}_t\left[e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right] \\
&\quad = \operatorname{Var}\left[\left(\pi_{\tau_{t+1}} - \pi_t\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right] - \operatorname{Var} E_t\left[\left(\pi_{\tau_{t+1}} - \pi_t\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right] \\
&\quad = \operatorname{Var}\left[\left(\pi_{\tau_{t+1}} - \pi_t\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right].
\end{aligned}$$

On the other hand,

$$\begin{aligned}
E \operatorname{Var}_t e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) &= E \operatorname{Var}_t\left[\pi_{\tau_{t+1}} - \pi_t + e^{-r(t+1)} E_t L_{t+1}\right] \\
&= E \operatorname{Var}_t\left[\pi_{\tau_{t+1}} - \pi_t\right] \\
&= \operatorname{Var}\left[\pi_{\tau_{t+1}} - \pi_t\right].
\end{aligned}$$

But $\hat{\pi}$ is a projection of π, and so

$$E \operatorname{Var}_t\left[e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right] \leq E \operatorname{Var}_t\left[e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right)\right]. \tag{3.13}$$

Equation (3.12) says that the regressor has the same conditional bias regardless of the presence of the control variate; equation (3.13) says that on average, the regressor with the control variate has lower conditional variance than the one without. Therefore, the control variate should improve the quality of the stopping times.

When the term Z_t is reintroduced, it is not clear that these properties are maintained. Although the $\mathscr{F}_t$-measurability of Z_t implies

$$E_t Z_t e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) = E_t Z_t \left(e^{-r\tau_{t+1}} g\left(\tau_{t+1}, X_{\tau_{t+1}}\right) - \left(\hat{\pi}_{\tau_{t+1}} - \hat{\pi}_t\right)\right),$$

so the conditional bias is still unchanged, the inequality corresponding to (3.13) may not hold and so we may not actually reduce variance by including the control variate. The point is that even though $\hat{\pi}$ is a projection of π, we cannot conclude that variance reduction occurs when we are restricted to the subset $[g(t, X_t) > 0]$. Nevertheless, it is reasonable to assume that there is variance reduction except perhaps when the option is deep out of the money so that the event $[g(t, X_t) > 0]$ occurs with very low probability. We have found in our numerical experiments that there is a (modest) improvement in using the control variate to estimate the stopping time; we continue to perform the regression using (3.11), retaining the indicator function as a heuristic.

3.4.2 The Algorithm

We establish the notation we use in the description of the algorithm. Let $\boldsymbol{\phi} = (\phi_0 = 1, \phi_1, \ldots, \phi_k)$ be a fixed set of basis functions which will be used in estimating the stopping time, as in (3.2). The fitted coefficients of this regression will be denoted $\boldsymbol{\alpha}_t \in \mathbb{R}^{k+1}$, for $t = 0, \ldots, T-1$. We will denote by $\boldsymbol{\theta}_t$ the complete set of parameters specifying the fitted extended MARS model (3.9) for the approximate value function $\hat{h}_t(\cdot) = \hat{h}_t(\cdot; \boldsymbol{\theta}_t)$, for $t = 1, \ldots, T$. The variables $Y(n)$,

$n = 1, \dots, N_1$, keep track of the cash flow along each path; the variables $cv(n)$ are the corresponding values of the control variate described in Section 3.4.1.

Algorithm 1 Phase One: Fit Stopping Times and Control Variate.

```
1: simulate $(X_0(n), \dots, X_T(n) : n = 1, \dots, N_1)$
2: $Y(n) \leftarrow g(T, X_T(n))$, for $n = 1, \dots, N_1$
3: $cv(n) \leftarrow 0$, for $n = 1, \dots, N_1$
4: for $t = T - 1, \dots, 0$ do
5:     $\boldsymbol{\theta}_t \leftarrow \text{argmin}_{\boldsymbol{\theta}} \sum_{n=1}^{N_1} \left( Y(n) - \hat{h}(X_{t+1}(n); \boldsymbol{\theta}) \right)^2$
6:     $cv(n) \leftarrow cv(n) + \hat{h}(X_{t+1}(n); \boldsymbol{\theta}_t) - E\left[ \hat{h}(X_{t+1}; \boldsymbol{\theta}_t) \mid X_t(n) \right]$, for $n = 1, \dots, N_1$
7:     $\boldsymbol{\alpha}_t \leftarrow \text{argmin}_{\boldsymbol{\alpha}} \sum_{n=1}^{N_1} 1_{[g(t, X_t(n)) > 0]} e^{-r} \left( \boldsymbol{\alpha}' \boldsymbol{\phi}(X_t(n)) - (Y(n) - cv(n)) \right)^2$
8:     for $n = 1, \dots, N_1$ do
9:         if $g(t, X_t(n)) > [\boldsymbol{\alpha}_t' \boldsymbol{\phi}(X_t(n))]_+$ then
10:            $Y(n) \leftarrow g(t, X_t(n)); cv(n) \leftarrow 0$
11:        else
12:            $Y(n) \leftarrow e^{-r} Y(n); cv(n) \leftarrow e^{-r} cv(n)$
13:        end if
14:    end for
15: end for
```

Algorithm 1 is the first phase of the pricing method where the stopping times and the control variate parameters are fit. After simulating the price paths and setting the value of the option at expiry to equal the payoff in lines 1-3, the algorithm proceeds backwards in time. Line 5 is where the MARS fitting algorithm is invoked; note that the "minimization" in this line is not a true minimization, due to the adaptive nature of the MARS fitting procedure. Line 6 updates the control variate for the stopping time as described in Section 3.4.1. Lines 7-14 are the Longstaff-Schwartz algorithm. Note that the fit in line 7 is trivial when $t = 0$ since we have assumed that X_0 is constant.

Algorithm 2 is the second phase wherein the lower and upper bounds on the option price are computed. New price paths are simulated (line 1), and the realized values are plugged in to the expressions for the martingale control variate (line 2) and the stopping strategy (line 3), both of which were computed during the first phase. Finally, the lower and upper bounds are computed on lines 4-5.

Algorithm 2 Phase Two: Compute Option Price Lower and Upper Bounds.

```
1: simulate *new* paths $(X_0(n), \ldots, X_T(n) : n = 1, \ldots, N_2)$
2: $\hat{\pi}_t(n) \leftarrow \sum_{s=1}^{t} e^{-rs} \left(\hat{h}(X_s(n); \boldsymbol{\theta}_s) - E\left[\hat{h}(X_s; \boldsymbol{\theta}_s) \mid X_{s-1}(n)\right]\right)$, for $n = 1, \ldots, N_2$, $t = 1, \ldots, T$
3: $\tau_0(n) \leftarrow T \wedge \min\{t = 0, \ldots, T-1 : g(t, X_t(n)) > \boldsymbol{\alpha}_t' \boldsymbol{\phi}(X_t(n))\}$, for $n = 1, \ldots, N_2$
4: $L_0 \leftarrow \frac{1}{N_2} \sum_{n=1}^{N_2} e^{-r\tau_0(n)} g(\tau_0(n), X_{\tau_0(n)}(n)) - \hat{\pi}_{\tau_0(n)}(n)$
5: $U_0 \leftarrow \frac{1}{N_2} \sum_{n=1}^{N_2} \max_{t=0,\ldots,T} \left(g(t, X_t(n)) - \hat{\pi}_t(n)\right)$
```

3.5 Numerical Examples

In this section, we describe how we have applied this algorithm to several multidimensional American option pricing problems, and we provide numerical results. In particular, we show how the conditional expectations (3.8) are computed.

All computations were performed using the R language (R Development Core Team 2005). The MARS fitting algorithm was originally developed in the S language by Hastie and Tibshirani; it was ported to R by Leisch et al. (2005). R is an interpreted language and thus can be fairly slow. Additionally, raw computation times may reflect details of implementation (e.g., R's garbage collection routines) and mask information that would be relevant in evaluating our algorithm. For this reason, we report the *ratio* of the computation time of the naïve estimator with that of our estimator for a fixed degree of accuracy, which we now explain.

In all experiments we fix the run lengths for Phase 1 and Phase 2 to 10,000 and 20,000 respectively, using common random numbers across experiments. We record the following quantities.

r_1 The time required in Phase 1 for both the LSM method and for MARS to fit the $\hat{L}_t$ functions.

r_2 The time required in Phase 2 to compute the MARS-based estimators of the lower and upper bounds.

$\tilde{r}_1$ The time required in Phase 1 for the LSM method alone.

$\tilde{r}_2$ The time required in Phase 2 to compute the naïve estimator of the lower bound.

s^2 An estimate of the variance of the MARS-based estimator of the lower bound.

$\tilde{s}^2$ An estimate of the variance of the naïve estimator of the lower bound.

$\hat{L}_0$ The MARS-based estimate of the lower bound.

We then compute the Phase 2 run lengths ($\tilde{n}$ and n for the naïve and MARS-based estimators respectively) required to achieve a confidence interval half-width for the lower bound that is approximately 0.1% of the lower bound estimate. Hence

$$\tilde{n} = \frac{1.96^2 \tilde{s}^2}{0.001^2 \hat{L}_0^2} \text{ and}$$
$$n = \frac{1.96^2 s^2}{0.001^2 \hat{L}_0^2}.$$

We then compute approximations for the computational time corresponding to these run lengths, viz

$$\tilde{R} = \tilde{r}_1 + \frac{\tilde{n}}{20,000}\tilde{r}_2 \text{ and}$$
$$R = r_1 + \frac{n}{20,000} r_2.$$

Finally, we report

$$\text{TR} = \tilde{R}/R$$

as an estimate of the speed-up factor (or time reduction) of the MARS-based estimator over the naïve estimator. We also report

$$\text{VR} = \tilde{s}^2/s^2$$

as the variance reduction factor. The former measure represents the true improvement in efficiency of the MARS-based estimator over the naïve estimator, while the latter measure indicates the variance reduction without adjustment for computation time.

In all examples, VR and TR are reported to two significant figures.

3.5.1 Asian Options

We begin by pricing Bermudan-Asian put options, under both the Black-Scholes and Heston (1993) models. In the Black-Scholes case, we have $(X_t : t = 0, \ldots, T) = ((S_t, A_t) : t = 0, \ldots, T)$, where S_0 is given and $S_1, \ldots, S_T$ are generated according to

$$S_t = S_{t-1} \exp\left(r - \frac{1}{2}\sigma^2 + \sigma W_t \right),$$

for independent standard normal variates $W_1, \ldots, W_T$. The average process $(A_t : t = 0, \ldots, T)$ is given by $A_0 = 0$ and, for $t \geq 1$,

$$A_t = \frac{1}{t}\sum_{s=1}^{t} S_s = \frac{1}{t}S_t + \frac{t-1}{t}A_{t-1} = \frac{1}{t}S_{t-1}\exp\left(r - \frac{1}{2}\sigma^2 + \sigma W_t \right) + \frac{t-1}{t}A_{t-1}. \tag{3.14}$$

The averaging dates are assumed to coincide with the possible exercise dates, excluding the date $t = 0$.

In continuous time, the Heston (1993) model is given by

$$\begin{aligned} dS_t &= \mu S_t dt + \sqrt{V_t} dW_t^{(1)}, \\ dV_t &= \kappa(\theta - V_t)dt + \sqrt{V_t}\sigma \left(\rho dW_t^{(1)} + \sqrt{1-\rho^2} dW_t^{(2)} \right), \end{aligned} \tag{3.15}$$

where $W^{(1)}$ and $W^{(2)}$ are independent Brownian motions. We approximate (3.15) in discrete time by applying the first-order Euler discretization to the logarithms of S_t and V_t. See Glasserman (2004, pp. 339–376) for details. This gives $(X_t : t = 0, \ldots, T) = (S_t, V_t, A_t : t = 0, \ldots, T)$ where S_0 and V_0 are given, and

$$\begin{aligned} V_t &= V_{t-1} \exp\left(\frac{\kappa\theta}{V_{t-1}} - \kappa - \frac{1}{2V_{t-1}}\sigma^2 + \frac{\rho\sigma W_t^{(1)} + \sqrt{1-\rho^2}\sigma W_t^{(2)}}{\sqrt{V_{t-1}}} \right) \\ S_t &= S_{t-1} \exp\left(r - \frac{1}{2}V_{t-1} + \sqrt{V_{t-1}} dW_t^{(1)} \right), \end{aligned} \tag{3.16}$$

where $W_1^{(1)}, \ldots, W_T^{(1)}, W_1^{(2)}, \ldots, W_T^{(2)}$ are independent standard normal variates. (The process $(A_t : t = 0, \ldots, T)$ is still given by (3.14).) The scheme (3.16) is not an exact discretization of (3.15); we ignore the discretization error and henceforth consider (3.16) to be the *true* dynamics of the underlying.

The payoff function of the Bermudan-Asian put is given by $g(0, \cdot) \equiv 0$ and

$$g(t, X_t) = (K - A_t)_+$$

for $t \geq 1$.

Let us consider the Heston case, as the Black-Scholes case is an easy specialization thereof. As mentioned in Section 3.3.1, we apply the MARS algorithm not to X_t but to a transformation of X_t which replaces S_t and V_t by their logarithms and A_t by the logarithm of the *geometric* average

$$\tilde{A}_t = \exp \frac{1}{t} \sum_{s=1}^{t} \log S_s.$$

We do not include the LDA directions in the Asian case. This yields an approximation

$$\begin{aligned} \hat{L}_t = \sum_{j=1}^{J_S} \alpha_{S,j} \Big(q_{S,j} \big[\log S_t - k_{S,j} \big] \Big)_+ + \sum_{j=1}^{J_V} \alpha_{V,j} \Big(q_{V,j} \big[\log V_t - k_{V,j} \big] \Big)_+ \\ + \sum_{j=1}^{J_A} \alpha_{A,j} \Big(q_{A,j} \big[\log \tilde{A}_t - k_{A,j} \big] \Big)_+ . \end{aligned}$$

The marginal conditional distributions of $\log S_t, \log V_t$, and $\log \tilde{A}_t$ given $\mathscr{F}_{t-1}$ are Gaussian, with mean and variance given by

$$
\begin{aligned}
E_{t-1} \log \begin{bmatrix} S_t \\ V_t \\ \tilde{A}_t \end{bmatrix} &= \begin{bmatrix} \log S_{t-1} + r - \frac{1}{2} V_{t-1} \\ \log V_{t-1} + \kappa\theta / V_{t-1} - \kappa - \frac{1}{2}\sigma^2 / V_{t-1} \\ \frac{1}{t}\Big((t-1) \log \tilde{A}_{t-1} + \log S_{t-1} + r - \frac{1}{2} V_{t-1} \Big) \end{bmatrix}, \\
\mathrm{Var}_{t-1} \log \begin{bmatrix} S_t \\ V_t \\ \tilde{A}_t \end{bmatrix} &= \begin{bmatrix} V_{t-1} \\ \sigma^2 / V_{t-1} \\ (1/t^2)\, V_{t-1} \end{bmatrix}.
\end{aligned}
\tag{3.17}
$$

(The full covariance matrix is irrelevant for our purpose.) This allows us to compute the conditional expectations $E_{t-1}\hat{L}_t$ easily.

Table 3.1 shows our computational results for Bermudan-Asian options. For all examples, we considered an option maturing in 6 months with monthly exercise/averaging dates; the annualized risk-free rate was $12r = .06$; the initial asset price was $S_0 = 100$. For the Heston examples, the *annualized* model parameters were $\kappa = 1.5, \sigma = .2, \theta = .36, \rho = -.75, V_0 = .4$. The stopping times were fit using the polynomials of degree up to 4 in S_t, A_t, and (for the Heston model) V_t, for $t = 1, \ldots, T-1$.

Table 3.1: Asian Option Results.

Model	K	Naïve L_0	MARS L_0	MARS U_0	VR	TR
BS ($\sigma = .3$)	95	2.77 (.07)	2.73 (.00)	2.78 (.00)	210	85
BS ($\sigma = .3$)	115	15.92 (.14)	15.86 (.01)	15.95 (.01)	230	44
BS ($\sigma = .6$)	95	7.88 (.15)	7.80 (.01)	7.94 (.01)	190	71
BS ($\sigma = .6$)	115	20.57 (.23)	20.48 (.02)	20.65 (.01)	230	56
Heston	95	5.04 (.11)	4.96 (.01)	5.06 (.01)	150	61
Heston	115	17.73 (.11)	17.66 (.01)	17.78 (.01)	200	50

Parenthesized values are 95% confidence interval half -widths. VR=Variance Reduction, TR=Time Reduction, defined at the top of Section 3.5.

In these examples, the reduction in variance is dramatic, ranging from about 150 times to 250 times variance reduction. Similarly, for an approximate 95% confidence interval with (relative) width .001, one needs to do about 50 times more work with the naïve estimator than with the one using the MARS-based control variate. Finally, observe that the closeness of the (MARS) estimates of L_0 and U_0 suggests that the stopping time found by the LSM algorithm is quite good.

3.5.2 Basket Options

Next, we consider options on baskets of d assets whose prices are given by $(X_t : t = 0, \ldots, T) = (S_t(i) : t = 0, \ldots, T; i = 1, \ldots, d)$. Specifically, we test call options on the maximum and on the average of the assets, which have respective payoff functions

$$g_{\max}(t,x) = \left(\vee_{i=1}^{d} x(i) - K\right)_+ , \quad g_{\text{avg}}(t,x) = \left(\frac{1}{d}\sum_{i=1}^{d} x(i) - K\right)_+ .$$

The underlying assets are assumed to follow the multidimensional Black-Scholes model, which is discretized as

$$S_t(i) = S_{t-1}(i) \exp\left(r - \delta - \frac{1}{2}\sigma_i^2 + \sigma_i W_t(i)\right), \tag{3.18}$$

for $i = 1, \ldots, d$, where $W_t = (W_t(1), \ldots, W_t(d))$ is a sequence of independent (in time) multivariate normal random variates with mean zero, unit variance, and a specified correlation matrix (see below). Here, δ is the dividend rate paid by each of the stocks per time step.

We take the annualized risk-free rate $12r$ to be .05, the dividend rate $12\delta = .1$, the annualized volatility to be $\sqrt{12}\sigma = .2$, the expiration to be 3 years, and the strike price to be $K = 100$. The dimension d of the problem takes the values

$d = 2, 3, 5, 10$. We test several values of the initial prices $(S_0(i))$, which are taken to be identical for $i = 1, \dots, d$. For the payoff function g_{avg}, we take the basis functions ϕ for fitting the stopping time τ to be the polynomials of degree up to two in the d asset prices. For the function $g_{\max}$, we take the basis functions to be the polynomials of degree up to two in the *order statistics* of the asset prices, which is similar to the choice of basis functions for such options in Longstaff and Schwartz (2001).

We divide each test further into three cases:

1. The assets' returns are uncorrelated,
2. The correlation between $W_t(i)$ and $W_t(j)$, for $i \neq j$, is a constant ρ, and
3. We randomly generate a correlation matrix for $(W_t(i), i = 1, \dots, d)$, $t = 1, \dots, T$, using the method of Marsaglia and Olkin (1984).

We test both the control variate based on MARS and the control variate based on LDA-MARS as in Section 3.3.2. For the LDA-MARS tests, we partition the sample paths at each time step t into three groups of approximately equal size corresponding to low, medium, and high values of $g(\tau_t, X_{\tau_t})$, and take the first two eigenvalues of the matrix (3.10), resulting in a total of nine LDA directions. (These are included in addition to, not instead of, the canonical directions.)

We apply the MARS and LDA-MARS fitting algorithms to the logarithm of S_t. The conditional distribution of $\log S_t$ given $\mathscr{F}_{t-1}$ is multivariate Gaussian with mean $\log S_{t-1} + r - \frac{1}{2}\sigma^2$, variance σ^2, and correlation matrix C which depends upon which of the three aforementioned cases we are in. Therefore, for a direction $a \in \mathbb{R}^d$, $\|a\| = 1$, the conditional distribution of $a' \log S_t$ given $\mathscr{F}_{t-1}$ is Gaussian with mean and variance

$$\begin{aligned} E_{t-1} a' \log S_t &= \sum_{i=1}^{d} a(i) \left(\log S_{t-1}(i) + r - \frac{1}{2}\sigma^2 \right) \\ \operatorname{Var}_{t-1} a' \log S_t &= \sigma^2 a' C a \end{aligned}$$

This allows us to compute the conditional expectations (3.8).

Table 3.2: Basket Option Results: Call on Average.

d	S_0	Naïve L_0	MARS L_0	LMARS L_0	LMARS U_0	MVR	MTR	LMVR	LMTR
Uncorrelated asset prices.									
2	90	1.98 (.07)	2.00 (.04)	1.99 (.01)	2.08 (.01)	3.2	2.8	51.0	43.0
2	100	4.90 (.10)	4.94 (.05)	4.93 (.01)	5.06 (.01)	4.4	3.9	73.0	59.0
3	90	1.08 (.05)	1.09 (.03)	1.10 (.01)	1.26 (.01)	2.2	1.8	18.0	15.0
3	100	3.61 (.07)	3.62 (.04)	3.63 (.01)	3.85 (.01)	3.2	2.9	32.0	29.0
5	90	0.39 (.02)	0.41 (.02)	0.42 (.01)	0.58 (.01)	1.4	1.1	6.4	4.3
5	100	2.32 (.05)	2.36 (.03)	2.37 (.01)	2.59 (.01)	2.4	1.9	19.0	15.0
10	90	0.05 (.01)	0.05 (.01)	0.05 (.00)	0.15 (.00)	1.0	0.5	2.3	0.2
10	100	1.18 (.03)	1.21 (.02)	1.25 (.01)	1.42 (.01)	1.9	1.3	13.0	8.3
Correlated asset prices ($\rho = .45$ for all asset pairs).									
2	90	3.08 (.09)	3.11 (.03)	3.09 (.01)	3.15 (.01)	7.3	6.8	83.0	78.0
2	100	6.39 (.13)	6.43 (.04)	6.38 (.01)	6.48 (.01)	11.0	9.7	120.0	110.0
3	90	2.61 (.08)	2.64 (.03)	2.62 (.01)	2.70 (.01)	6.0	5.1	67.0	56.0
3	100	5.77 (.12)	5.82 (.04)	5.80 (.01)	5.91 (.01)	8.6	7.4	95.0	83.0
5	90	2.15 (.07)	2.25 (.03)	2.24 (.01)	2.33 (.01)	4.6	3.4	60.0	48.0
5	100	5.27 (.10)	5.32 (.04)	5.30 (.01)	5.41 (.01)	6.9	5.8	91.0	75.0
10	90	1.77 (.06)	1.92 (.03)	1.96 (.01)	2.08 (.01)	4.0	2.8	35.0	23.0
10	100	4.71 (.10)	4.87 (.04)	4.90 (.01)	5.02 (.01)	7.0	5.7	80.0	64.0
Correlated asset prices (random correlation matrix).									
5	90	0.07 (.01)	0.08 (.01)	0.08 (.01)	0.15 (.00)	2.5	0.9	3.7	0.5
5	100	0.85 (.02)	0.88 (.01)	0.89 (.01)	1.03 (.01)	3.2	2.5	11.0	8.1
10	90	0.09 (.01)	0.09 (.01)	0.10 (.01)	0.21 (.01)	1.2	0.6	2.9	0.3
10	100	1.38 (.03)	1.40 (.02)	1.42 (.01)	1.62 (.01)	2.1	2.0	11.0	9.3

Parenthesized values are 95% confidence interval half-widths. MVR/LMVR = MARS/LMARS variance reduction. MTR/LMTR = MARS/LMARS time reduction, defined at the top of Section 3.5.

For the call on the average, the variance reduction using LDA-MARS is quite good, resulting in a speed-up factor of between about 5 and 50 for both the uncorrelated case and the randomly correlated case, and between about 25 and 110 for the positively correlated case. There is some degradation of performance as the dimension increases from 2 to 10. We also observe that the variance reduction is much greater for options at-the-money than out-of-the-money.

It is natural to expect that LDA-MARS should perform significantly better than MARS for an option on the average of stocks, as there is one linear direction (namely, $a = (1, \ldots, 1)$) that is likely to capture much of the variation in the value function. It is also plausible that the effect of the control variate is stronger when the assets are positively correlated, and that the degradation with dimension is smaller in that case as well, since under this correlation structure much of the variance of the assets' returns is driven by a single factor. Both of these observations are borne out in the results.

Table 3.3: Basket Option Results: Call on Max, Uncorrelated Asset Prices.

d	S_0	Naïve L_0	MARS L_0	LMARS L_0	LMARS U_0	MVR	MTR	LMVR	LMTR
2	90	7.92 (.16)	8.05 (.03)	8.08 (.03)	8.40 (.02)	23	18	37	30
2	100	13.77 (.20)	13.88 (.05)	13.90 (.03)	14.46 (.03)	18	17	40	37
2	110	21.27 (.23)	21.33 (.06)	21.33 (.04)	22.09 (.04)	14	11	39	33
3	90	11.15 (.18)	11.20 (.05)	11.17(.04)	11.92 (.04)	14	10	17	13
3	100	18.58 (.23)	18.57 (.07)	18.60 (.06)	20.01 (.05)	11	8.9	16	12
3	110	27.42 (.26)	27.42 (.09)	27.43 (.07)	29.22 (.06)	8.7	6.9	16	12
5	90	16.27 (.21)	16.46 (.08)	16.46 (.08)	18.18 (.07)	6.4	4.8	7.5	5.1
5	100	25.83 (.25)	25.98 (.11)	25.97 (.10)	28.79 (.08)	5.5	4.3	6.8	5.0
5	110	36.46 (.28)	36.58 (.13)	36.53 (.11)	40.20 (.10)	4.8	3.7	6.5	4.7
10	90	25.70 (.24)	25.85 (.12)	25.87 (.12)	29.20 (.11)	3.9	2.9	4.3	3.1
10	100	37.73 (.27)	39.74 (.14)	37.94 (.13)	42.14 (.12)	3.6	2.7	4.2	3.1
10	110	50.19 (.30)	50.47 (.16)	50.45 (.15)	55.25 (.13)	3.6	2.7	4.1	3.0

Parenthesized values are 95% confidence interval half -widths. MVR/LMVR = MARS/LMARS variance reduction. MTR/LMTR = MARS/LMARS time reduction, defined at the top of Section 3.5.

Table 3.4: Basket Option Results: Call on Max, Correlated Asset Prices.

d	S_0	Naïve L_0	MARS L_0	LMARS L_0	LMARS U_0	MVR	MTR	LMVR	LMTR
$\rho = .45$ for all asset pairs.									
2	90	7.11 (.16)	7.18 (.04)	7.19 (.03)	7.68 (.03)	16	13	28	22
2	100	12.20 (.20)	12.28 (.05)	12.35 (.04)	13.08 (.03)	15	14	31	28
2	110	19.02 (.24)	19.00 (.06)	19.03 (.04)	20.00 (.04)	15	12	35	25
3	90	9.26 (.19)	9.27 (.06)	9.30 (.05)	10.47 (.04)	11	8.1	15	11
3	100	15.33 (.23)	15.35 (.07)	15.38 (.06)	17.04 (.05)	11	8.4	17	13
3	110	22.97 (.27)	22.95 (.08)	22.97 (.07)	24.98 (.06)	11	8.3	17	13
5	90	12.33 (.21)	12.46 (.08)	12.49(.08)	14.53 (.07)	6.9	5.3	8.1	5.8
5	100	19.65 (.26)	19.80 (.10)	19.84 (.09)	22.94 (.09)	6.7	5.1	8.0	5.8
5	110	28.36 (.30)	28.52 (.12)	28.55 (.10)	32.42 (.10)	6.8	5.1	9.0	6.4
10	90	17.46 (.26)	17.64 (.11)	17.64(.11)	20.86(.12)	5.5	3.8	6.0	3.7
10	100	26.43 (.31)	26.58 (.13)	26.67 (.12)	31.20 (.13)	5.4	3.6	6.2	4.8
10	110	36.50 (.35)	36.74 (.15)	36.69 (.14)	42.21 (.15)	5.5	4.1	6.5	4.7
random correlation matrix									
5	90	15.78 (.20)	15.89 (.06)	15.88 (.06)	17.16 (.06)	9.7	7.4	10	7.6
5	100	25.45 (.23)	25.61 (.08)	25.63 (.07)	27.70 (.07)	8.8	6.8	9.4	6.8
5	110	36.35 (.25)	36.52 (.09)	36.53 (.09)	39.25 (.07)	7.7	5.9	8.5	6.0
10	90	23.55 (.24)	23.79 (.12)	23.79 (.11)	27.02 (.11)	4.3	2.9	4.9	3.1
10	100	35.10 (.28)	35.35 (.14)	35.41 (.13)	39.69 (.12)	4.0	3.4	4.7	3.6
10	110	47.23 (.30)	47.56 (.15)	47.53 (.14)	52.56 (.13)	3.9	3.0	4.5	3.2

Parenthesized values are 95% confidence interval half -widths. MVR/LMVR = MARS/LMARS variance reduction. MTR/LMTR = MARS/LMARS time reduction, defined at the top of Section 3.5.

The results are somewhat less dramatic for the case of the option on the maximum. This is most likely due to the fact that the payoff function g_{max} is highly non-separable, so the fitted functions $\hat{L}$ are poor approximations for the true value functions L. In fact, not only is g_{max} non-separable, but it cannot even be represented exactly in the form (3.9). Thus, even when LDA directions are used, and even in the correlated assets case, the performance degrades quickly to a variance reduction factor of only around 2 or 3 as the dimension increases. Still, the method seems to be able to provide about a threefold decrease in computation time even in this case. We also observe that the effects of correlation are much less noticable for the call on the max than for the call on the average.

The first nine rows of the first panel of Table 3.3 may be compared with Table 2 of Andersen and Broadie (2004). Our results (using LDA-MARS) include confidence intervals that are approximately twice the width of the ones reported in Andersen and Broadie (2004), although we use 20,000 simulation trials to their 2,000,000 trials. In order to get confidence intervals of the same order, we would need to use approximately 80,000 trials – still quite a bit fewer than 2,000,000. On the other hand, the "duality gaps" between the upper and lower bounds are much tighter in Andersen and Broadie (2004) than in our study. This stands to reason; our upper bounds are wholly reliant on the approximation $\hat{\pi}$ for π; in contrast, they compute π explicitly by running additional simulation trials.

3.5.3 Barrier Options

Finally we test our method on a variety of barrier options: the up-and-out call, the up-and-out put, and the down-and-out put, all on a single asset. Unlike a vanilla Bermudan call, a Bermudan up-and-out call on an asset that does not pay dividends may have an optimal exercise policy other than the trivial one $\tau_0 = T$. Again, we test both the Black-Scholes and the Heston models.

Although it would be possible to accommodate the path dependence of barrier options by expanding the state space, we adopt a different approach. Let $\mathscr{B} \subset \mathbb{R}^d$ denote the region in which the option is knocked out. Assume the payoff function g satisfies $g(\cdot, x) \equiv 0$ for all $x \in \mathscr{B}$. For each $t = 0, \ldots, T$, let ν_t be the first hitting time of $\mathscr{B}$ between t and T, or $T+1$ if there is no such hitting time, i.e.,

$$\nu_t = \inf\{s = t, \ldots, T+1 : (s, X_s) \in \{t, \ldots, T\} \times \mathscr{B} \cup \{T+1\} \times \mathbb{R}^d\}.$$

We now redefine our value function to be

$$Q_t = \sup_{\tau \in \mathcal{T}(t)} E_t \left[e^{-r(\tau \wedge \nu_t - t)} g\left(\tau \wedge \nu_t, X_{\tau \wedge \nu_t}\right) \right]. \tag{3.19}$$

The stopping times τ_t^* solving (3.19) satisfy

$$\begin{aligned} &\tau_T^* \wedge \nu_T = T, \\ &\tau_t^* \wedge \nu_t = \begin{cases} t & \text{if } \nu_t = t \text{ or if } g(t, X_t) \geq e^{-r} E_t Q_{t+1}, \\ \tau_{t+1}^* & \text{otherwise,} \end{cases} \end{aligned}$$

for $t = 0, \ldots, T-1$. The suboptimal stopping times $(\tau_t : t = 0, \ldots, T)$ are defined analogously to those in Section 3.2.1. In this setting the analogous martingale π satisfies

$$\pi_{\tau_t \wedge \nu_t} - \pi_t = e^{-r\tau_t \wedge \nu_t} g\left(\tau_t \wedge \nu_t, X_{\tau_t \wedge \nu_t}\right) - e^{-rt} L_t,$$

similar to Proposition 3.1, and we have

$$Q_0 \leq E \max_{t=0,\ldots,T} \left[g\left(t \wedge \nu_0, X_{t \wedge \nu_0}\right) - \pi_{t \wedge \nu_0} \right].$$

In other words, the martingale π evaluated only as far as the hitting time of the knock-out region, both for computing the control variate and the upper bound.

This leads to Algorithms 3 and 4, which are modifications of Algorithms 1 and 2, respectively. The only difference between Algorithms 1 and 3 occurs on line 9, which in the barrier option case says to zero out the cash flow and the control variate upon exercise *or knockout*. Algorithm 4 differs from Algorithm 2 in that the exercise time τ_0 is replaced with the minimum of the exercise time and knock out time $\tau_0 \wedge \nu_0$.

Algorithm 3 Phase One (Barrier Option): Fit Stopping Times and Control Variate.

```
1:  simulate $(X_0(n), \ldots, X_T(n) : n = 1, \ldots, N_1)$
2:  $Y(n) \leftarrow g(T, X_T(n))$, for $n = 1, \ldots, N_1$
3:  $cv(n) \leftarrow 0$
4:  for $t = T-1, \ldots, 0$ do
5:      $\boldsymbol{\theta}_t \leftarrow \text{argmin}_{\boldsymbol{\theta}} \sum_{n=1}^{N_1} \left(Y(n) - \hat{h}(X_{t+1}(n); \boldsymbol{\theta})\right)^2$
6:      $cv(n) \leftarrow cv(n) + \hat{h}(X_{t+1}(n); \boldsymbol{\theta}_t) - E\left[\hat{h}(X_{t+1}; \boldsymbol{\theta}_t) \mid X_t(n)\right]$, for $n = 1, \ldots, N_1$
7:      $\boldsymbol{\alpha}_t \leftarrow \text{argmin}_{\boldsymbol{\alpha}} \sum_{n=1}^{N_1} 1_{[g(t, X_t(n)) > 0]} e^{-r} \left(\boldsymbol{\alpha}' \boldsymbol{\phi}(X_t(n)) - (Y(n) - cv(n))\right)^2$
8:      for $n = 1, \ldots, N_1$ do
9:          if $g(t, X_t(n)) > [\boldsymbol{\alpha}_t' \boldsymbol{\phi}(X_t(n))]_+$ or $X_t(n) \in \mathscr{B}$ then
10:             $Y(n) \leftarrow g(t, X_t(n)); cv(n) \leftarrow 0$
11:         else
12:             $Y(n) \leftarrow e^{-r} Y(n); cv(n) \leftarrow e^{-r} cv(n)$
13:         end if
14:     end for
15: end for
```

Algorithm 4 Phase Two (Barrier Option): Compute Option Price Lower and Upper Bounds.

```
1:  simulate new paths $({}_0(n), \ldots, X_T(n) : n = 1, \ldots, N_2)$
2:  $\hat{\pi}_t(n) \leftarrow \sum_{s=1}^{t} e^{-rs} \left(\hat{h}(X_s(n); \boldsymbol{\theta}_s) - E\left[\hat{h}(X_s; \boldsymbol{\theta}_s) \mid X_{s-1}(n)\right]\right)$, for $n = 1, \ldots, N_2$, $t = 1, \ldots, T$
3:  $\tau_0(n) \leftarrow T \wedge \min\{t = 0, \ldots, T-1 : g(t, X_t(n)) > \boldsymbol{\alpha}_t' \boldsymbol{\phi}(X_t(n))\}$, for $n = 1, \ldots, N_2$
4:  $\nu_0(n) \leftarrow (T+1) \wedge \min\{t = 0, \ldots, T : X_t(n) \in \mathscr{B}\}$
5:  $L_0 \leftarrow \frac{1}{N_2} \sum_{n=1}^{N_2} e^{-r(\tau_0(n) \wedge \nu_0(n))} g((\tau_0(n) \wedge \nu_0(n)), X_{(\tau_0(n) \wedge \nu_0(n))}(n)) - \hat{\pi}_{(\tau_0(n) \wedge \nu_0(n))}(n)$
6:  $U_0 \leftarrow \frac{1}{N_2} \sum_{n=1}^{N_2} \max_{t=0,\ldots,\nu_0(n)} \left(g(t, X_t(n)) - \hat{\pi}_t(n)\right)$
```

In our numerical experiments on Bermudan barrier options, the underlying dynamics of $(X_t : t = 0, \ldots, T) = (S_t : t = 0, \ldots, T)$ or $(X_t : t = 0, \ldots, T) = ((S_t, V_t) : t = 0, \ldots, T)$ follow (3.18) or (3.16) accordingly, and the appropriate parameters of the one-step conditional distributions are still given by (3.17) (without the A_t term). The model parameters are as described in Section 3.5.1. Tables 3.5 and 3.6 report the computational results for barrier options in the Black-Scholes and Heston models, respectively.

The performance of the control variate for the barrier option examples seems to have huge variability, with variance reduction factors ranging from 8.5 to 350 just within the Black-Scholes cases. Why is there such a discrepancy in the quality of the algorithm between these two examples? The cases for which the control variate is very successful are the "up-and-out" puts, which knock out when the option is deep out-of-the-money. The less successful cases are the "up-and-out" call and the "down-and-out" put, which knock out in-the-money and as such have discontinuous payoff functions. Indeed, this discontinuity is notorious for causing headaches among traders, especially in currency markets, who must hedge these options. Our problem here is that MARS has difficulty fitting these functions as well as it fits the smoother "up-and-out" put value functions. We believe that with a little manual tweaking we could get the performance for the discontinuous cases to improve significantly. However, in the spirit of having a fully automated procedure we have not pursued this line of inquiry. An interesting future research project would be to develop a version of MARS that is more robust to discontinuities in the target function.

Table 3.5: Barrier Option Results (Black-Scholes, $\sigma = .3$).

Stgy.	K	B	Naïve L_0	MARS L_0	MARS U_0	VR	TR
Call	95	130	13.38 (.16)	13.36 (.04)	14.11 (.04)	18.0	15.0
Call	115	130	3.19 (.06)	3.19 (.02)	3.44 (.02)	8.5	7.3
Put	95	70	6.61 (.11)	6.7 (.03)	7.25 (.03)	14.0	13.0
Put	115	70	18.34 (.19)	18.45 (.02)	19.30 (.04)	79.0	53.0
Put	95	130	6.98(.14)	7.09 (.01)	7.13 (.01)	180.0	140.0
Put	115	130	17.72 (.20)	17.86 (.01)	17.92 (.01)	350.0	140.0

Parenthesized values are 95% confidence interval half -widths. VR=Variance Reduction, TR=Time Reduction, defined at the top of Section 3.5.

Table 3.6: Barrier Option Results (Heston).

Stgy.	K	B	Naïve L_0	MARS L_0	MARS U_0	VR	TR
Call	95	130	14.06 (.15)	13.98 (.05)	14.82 (.05)	9.9	8.5
Call	115	130	3.36 (.06)	3.33 (.03)	3.66 (.02)	5.0	4.3
Put	95	70	8.95 (.12)	8.99 (.05)	9.82 (.04)	5.2	4.5
Put	115	70	22.12 (.20)	22.20 (.06)	23.72 (.06)	9.6	7.9
Put	95	130	12.93(.24)	13.10(.02)	13.16 (.02)	170.0	120.0
Put	115	130	22.04 (.33)	22.34 (.03)	22.51 (.02)	170.0	110.0

Parenthesized values are 95% confidence interval half -widths. VR=Variance Reduction, TR=Time Reduction, defined at the top of Section 3.5.

3.6 Conclusion

We have presented a new, automated procedure for finding control variates for the American option pricing problem. The key advantages of our method are its degree of applicability to many option types and stochastic processes, without requiring much additional implementation overhead, and its use of off-the-shelf software. Our method works extremely well for problems of moderate dimension (up to about 5), and for problems where much of the variability of the underlying processes can be explained with a moderate number of parameters. Moreover, the method can “discover” such structure automatically as a result of using an adaptive fitting procedure.

A possible area of future research would be to apply this technique in conjunction with quasi-Monte Carlo methodology. This would likely result in even greater variance reductions, although that remains to be seen. The good news is that the overall procedure would not change in any substantive way. Finally, this paper suggests that there is promise in applying techniques from the (vast) statistical data mining literature to the American option pricing problem. This is a direction we hope to continue to pursue.

Chapter 4

Deterministic and Stochastic Root Finding in One Dimension for Increasing Convex Functions

4.1 Introduction

Suppose $h_* : [a, b] \to \mathbb{R}$ is non-decreasing and convex, with known upper bound c on the left derivative at b. Suppose further that h_* is known to have a unique root in $[a, b]$. In this paper, we discuss efficient algorithms for finding an approximate root of h_*. Our notion of an approximate root is different from the usual one in that we are concerned only with finding a point $\bar{x}$ satisfying $|h_*(\bar{x})| \leq \delta$, where $\delta > 0$ is the *tolerance parameter*; proximity to the actual root of h_* is irrelevant. Such an $\bar{x}$ is called a δ-root. We consider both the cases where h_* can be evaluated exactly and where h_* must be evaluated by stochastic simulation. In the latter case, we suppose that confidence interval estimates of the function values are available.

Our interest in this form of the root-finding problem arises from the problem of pricing American options (eg, (Duffie 2001), (Glasserman 2004)). In approaches for American option pricing which combine simulation with backward stochastic dynamic programming (eg, (Carriere 1996), (Grant, Vora, and Weeks 1996), (Longstaff and Schwartz 2001)) one must classify all points in the state space as points where it is preferable to exercise the option immediately or as points where it is preferable to continue to hold the option. If the state space is one dimensional, then under quite general conditions this amounts to finding the root of a particular convex function. This function must be computed via simulation. The absolute value of this function at a point x indicates the financial loss associated with making the wrong decision in state x.

A standard approach to stochastic root finding is stochastic approximation, which has an enormous literature; see, e.g., (Kushner and Yin 2003). However, stochastic approximation is fundamentally a *search* procedure, and does not provide a probabilistic guarantee. This is also true of Simultaneous-Perturbation Stochastic Approximation (Spall 2003). Chen and Schmeiser (Chen and Schmeiser 2001) develop retrospective root finding algorithms that progressively narrow in on a root, and, like stochastic approximation, the theory is directed towards proving convergence in a limiting sense, rather than offering finite-time performance guarantees. This is also true of the root finding procedures developed in (Pasupathy and Schmeiser 2003; Pasupathy and Schmeiser 2004).

Our work is related to algorithms developed for the root finding problem with exact function evaluations, especially (Gross and Johnson 1959), but also of note are (Potra 1994), in which a quadratic rate of convergence is established for a root finding algorithm applied to a subclass of smooth convex functions, and (Rote 1992), which surveys and analyzes the sandwich algorithm for shrinking a polygonal envelope of a convex function. None of the papers we have seen attempt to identify a δ-root, instead adopting other error measures.

Therefore, the first main contribution of our work, namely the development and analysis of an algorithm for δ-root finding with *exact* function evaluations, shares the problem context of these papers, but we work with both a different error measure and a different class of functions. The second main contribution extends this algorithm and its analysis to the case of *inexact* function evaluations. We provide a bound on the number of iterations required in each case, where in the stochastic case the bound holds with high probability. In particular, this bound is an improvement over that of our algorithm's natural competitor, the bisection algorithm, by a constant factor.

The remainder of this paper contains two main sections. Section 4.2 deals with the deterministic case and §4.3 deals with the stochastic case. For the deterministic case, we begin by defining notation and briefly recalling the performance guarantee one can achieve with the simple bisection algorithm. Section 4.2.1 defines envelope functions that sandwich h_* and discusses how these envelopes can be used to guide a root-finding procedure. In §4.2.2 we state and prove the performance guarantee.

We begin §4.3 by discussing the natural extension of bisection to the stochastic case. Section 4.3.1 describes how to construct envelopes in the stochastic setting. Section 4.3.2 provides a performance guarantee assuming that the interval estimates have probabilty zero of error. In §4.3.3 we relax that assumption and introduce an adaptive algorithm for determining the confidence levels with which the interval estimates must be generated. We provide a heuristic performance improvement in §4.3.4 based upon the idea of completely solving the root-finding problem for a decreasing sequence of tolerance parameters.

4.2 Deterministic δ-root finding

In this section, we discuss the version of the problem in which exact evaluations of the unknown function h_* are available. We also establish notation and terminology to use throughout the paper.

For constants $a < b$ and $c > 0$, let $\mathcal{H}_0 = \mathcal{H}_0(a, b, c)$ be the set of all convex, non-decreasing functions $h : [a, b] \to \mathbb{R}$ that are continuous at a and b, possess a unique zero in $[a, b]$, and whose left derivative at b is bounded above by c.

The algorithms discussed in this paper evaluate h_* at a sequence of points until a stopping condition is reached. For $j \geq 1$, let x_j be the point at which h_* is evaluated in the j^{th} iteration and let $y_j = h_*(x_j)$. For $k \geq 1$, let $\mathbf{x}_k = (x_1, \ldots, x_k)$ and $\mathbf{y}_k = (y_1, \ldots, y_k)$. Denote by $x^k_{[1]}, \ldots, x^k_{[k]}$ and $y^k_{[1]}, \ldots, y^k_{[k]}$ the order statistics of $\mathbf{x}_k$ and $\mathbf{y}_k$, respectively. We typically omit the superscripts on these quantities, as the relevant iteration is usually clear from the context. The fact that h_* is known to be non-decreasing implies that the order statistic notation is consistent, i.e., $h_*(x_{[j]}) = y_{[j]}$, for $j = 1, \ldots, k$. Let $\mathcal{H}_k$ denote the set of all functions $h \in \mathcal{H}_0$ such that $h(x_j) = y_j$, for $j = 1, \ldots, k$. In particular, we have $h_* \in \mathcal{H}_k$ for $k \geq 0$.

Perhaps the most obvious choice of algorithm for locating a δ-root is bisection, e.g., (Press, Flannery, Teukolsky, and Vetterling 1992, pp. 350–354). Let $\mathcal{B}_k$ denote the *bracket* after k iterations of the algorithm, defined to be the smallest interval whose endpoints are in $\{a, x_1, \ldots, x_k, b\}$ that is known to contain the root of h_*. After k iterations one next evaluates h_* at the midpoint of $\mathcal{B}_k$. Since the width of the bracket decreases by a factor of 2 at each iteration (i.e., the sequence of bracket widths converges to zero linearly at rate 2), the bracket width after $k \geq 0$ iterations is $2^{-k}(b - a)$. The fact that the growth rate of h_* on any interval is *a priori* bounded above by c guarantees that once the bracket width is no greater than $2\delta/c$, the midpoint of the bracket is a δ-root. Thus, the algorithm can be designed to terminate after $\lceil \log_2((b - a)c/\delta) - 1 \rceil$ function evaluations or fewer.

Other derivative-free one-dimensional root finding algorithms, as described, e.g., in (Potra 1994), (Press, Flannery, Teukolsky, and Vetterling 1992, pp. 354–362), exhibit super-linear convergence, but only when the function is known to satisfy certain smoothness conditions and the root has already been sufficiently well-bracketed. In such algorithms, it is the sequence of bracket widths which converges super-linearly in accordance with the goal of finding a point close to the true root. In contrast, our algorithm measures progress according to a sequence of quantities $(q_k : k \geq 2)$, described in §4.2.2, which are specially designed to measure progress towards finding a δ-root. The main result of this section is that this sequence converges linearly to zero *at rate 3* for the entire class of functions $\mathcal{H}_0$ until such time as a δ-root is found.

4.2.1 Envelope functions

For the remainder of §2, we suppose the endpoints of the initial bracket $[a, b]$ are evaluated in the first two iterations, so that $(x_1, x_2) = (a, b)$. Although not strictly necessary, this will simplify notation somewhat. Suppose further that $h(a) < -\delta$ and $h(b) > \delta$, for otherwise there is no work to be done in finding a δ-root.

Once $k \geq 2$ points have been evaluated, one can derive piecewise linear functions u_k and l_k that provide tight bounds on the functions in $\mathcal{H}_k$. We name these functions the upper and lower envelopes, respectively. The upper envelope u_k is simply the linear interpolant of the points (x_j, y_j), for $j = 1, \ldots, k$. The lower envelope l_k is constructed by extending the line segment joining consecutive points $(x_{[j]}, y_{[j]}), (x_{[j+1]}, y_{[j+1]})$ to the left of $x_{[j]}$ (for $j = 2, \ldots, k$) and to the right of $x_{[j+1]}$ (for $j = 1, \ldots, k-1$), extending a horizontal ray to the right of $(a, h_*(a))$, and extending a ray with slope c to the left of $(b, h_*(b))$. The pointwise maximum of these $2k - 2$ rays is the lower envelope; see Figure 4.1. This same construction is given in (den Boef and den Hertog 2007) and (Rote 1992). In §4.3.1 we extend this procedure to the case where function evaluations are inexact, and prove (in the more general setting) that the resulting bounds are tight in the sense that $u_k(x) = \sup_{h\in\mathcal{H}_k} h(x)$ and $l_k(x) = \inf_{h\in\mathcal{H}_k} h(x)$.

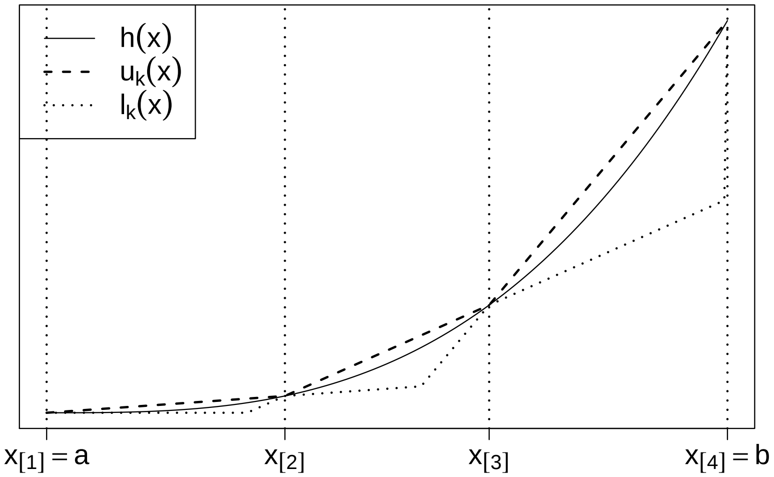


Figure 4.1: Envelope functions.

Observe that both u_k and l_k may be constant on certain intervals. Accordingly, for any continuous function $f : [a,b] \to \mathbb{R}$ we denote by $f^{\leftarrow}$ and $f^{\rightarrow}$ the left and right inverses of f, respectively, given by

$$f^{\leftarrow}(y) = \inf\{x \in [a,b] : f(x) = y\}$$
$$f^{\rightarrow}(y) = \sup\{x \in [a,b] : f(x) = y\}.$$

If $x \in \mathcal{B}_k = [x_{[i]}, x_{[i+1]})$ is the root of h_*, then

$$x_{[i]} \le u_k^{\leftarrow}(0) \le x \le l_k^{\rightarrow}(0) \le x_{[i+1]}.$$

In other words, the envelopes allow us to deduce an interval that is smaller than the bracket in which to search for δ-roots.

We can also use the envelopes to determine a stopping condition for the algorithm. In particular, if

$$l_k^{\rightarrow}(-\delta) \le x \le u_k^{\leftarrow}(\delta), \tag{4.1}$$

then

$$-\delta \le l_k(x) \le u_k(x) \le \delta,$$

implying that x must be a δ-root of h_*. It is possible that (4.1) may be met for some x even if h_* has not actually been evaluated at a δ-root, as illustrated in Figure 4.2.

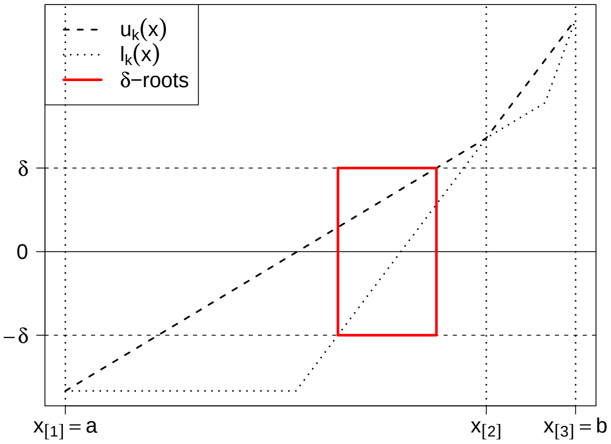


Figure 4.2: Stopping condition.

4.2.2 Reductions in potential

In much the same spirit as (den Boef and den Hertog 2007), we use the information contained in u_k and l_k to extract information to guide our search procedure, including to determine the sequence of points at which to evaluate h_*. Before describing how this is done, we introduce some more notation.

The lower envelope l_k is piecewise linear but not necessarily convex. Indeed, l_k is locally *concave* at each ordinate x_j. Denote by $z^-_{[j]}$ and $z^+_{[j]}$, respectively, the left and right derivatives of l_k at $x_{[j]}$ (take $z^-_{[1]} = 0$ and $z^+_{[k]} = c$). Let $\mathbf{z}_k = (z^-_{[j]}, z^+_{[j]} : j = 1, \ldots, k)$. The dependence of these quantities upon the iteration k is usually suppressed in the notation; if we need to make the iteration k explicit, we write $z^{\pm}_{[j]}(k)$.

As we will see in §4.3.1, the quantities $z^-_{[j]}$ and $z^+_{[j]}$ provide tight bounds on the subgradient of all functions $h \in \mathcal{H}_k$ at $x_{[j]}$; moreover, they turn out to be helpful in the construction of the lower envelope in the stochastic setting. For the time being, we simply remark that $\mathbf{z}_k$ can be computed using only $\mathbf{x}_k$ and $\mathbf{y}_k$, as well as the known constants b and c. We note that the order statistic notation is consistent here as well, as $z^-_{[j]} \leq z^-_{[j+1]}$ and $z^+_{[j]} \leq z^+_{[j+1]}$, for $j = 1, \dots, k-1$.

We now discuss the sequence of quantities $(q_k : k \geq 2)$ which tracks the progress of the algorithm. This in turn leads us to a particular sampling strategy. Given $k \geq 2$, suppose $\mathcal{B}_k = [x_{[i]}, x_{[i+1]})$. Define the *potential* q_k by:

$$q_k = y^2_{[i+1]}(z^+_{[i+1]}\Delta_k^{-1} - 1), \tag{4.2}$$

where

$$\Delta_k = \frac{y_{[i+1]} - y_{[i]}}{x_{[i+1]} - x_{[i]}}$$

is the slope of u_k on $\mathcal{B}_k$.

There is a geometric interpretation to q_k. Namely, it is twice the product of the slope $z^+_{[i+1]}$ with the area of the triangle formed by the x-axis, the graph of u_k, and the ray to the left of $x_{[i+1]}$ appearing in the construction of the lower envelope.

The potential q_k plays a role in our algorithm analogous to that played by the width of $\mathcal{B}_k$ in bisection. Lemma 4.1 shows that once q_k is sufficiently small, the stopping condition (4.1) holds for a particular value of x.

Lemma 4.1. *Let*

$$\hat{x} = x_{[i+1]} - \frac{y_{[i+1]}}{z^+_{[i+1]}}.$$

Then we have

$$0 \leq l_k(\hat{x}) \leq u_k(\hat{x}) \leq \frac{1}{2}\sqrt{q_k}. \qquad (4.3)$$

In particular, if $q_k \leq 4\delta^2$ *then* $\hat{x}$ *is a* δ*-root.*

Proof. The first two inequalities in (4.3) follow directly from the definition of $\hat{x}$ and of the envelope functions. For the proof of the third and final inequality, consider Figure 4.3. By the similarity of the triangles ABE and ACD, we have

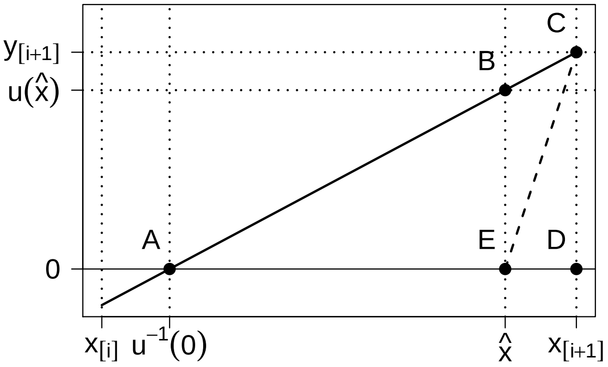


Figure 4.3: Similar triangles in proof of Lemma 4.1. Here, slope($\overline{AC}$) = Δ and slope($\overline{CE}$) = $z^+_{[i+1]}$.

$$u(\hat{x}) = y_{[i+1]} \cdot \left(1 - \frac{\Delta_k}{z^+_{[i+1]}}\right) = y_{[i+1]}\frac{z^+_{[i+1]}\Delta_k^{-1} - 1}{z^+_{[i+1]}\Delta_k^{-1}} = y_{[i+1]}\frac{q_k}{q_k + y^2_{[i+1]}}.$$

For any fixed value of q_k, the latter expression is maximized at $y_{[i+1]} = \sqrt{q_k}$. This proves (4.3). □

We now consider how much of a reduction in potential is achieved at each iteration. The key idea is to pick a point x_* at which to sample h_* which equates upper bounds on the potential in the two cases $h_*(x_*) > 0$ and $h_*(x_*) < 0$. To that end, let

$$x_* = x_{[i+1]} - y_{[i+1]}\frac{\Delta_k^{-1}}{2\sqrt{z^+_{[i+1]}\Delta_k^{-1} - 1}}. \qquad (4.4)$$

Lemma 4.2. *If $\mathcal{B}_k = [x_{[i]}, x_{[i+1]})$ and the $(k+1)^{th}$ sampling point is given by* (4.4)*, then*

$$\frac{q_{k+1}}{q_k} \leq \frac{\sqrt{z^+_{[i+1]}\Delta_k^{-1}} - 1}{2z^+_{[i+1]}\Delta_k^{-1} + \sqrt{z^+_{[i+1]}\Delta_k^{-1}} - 1}. \tag{4.5}$$

Proof. We consider two separate cases depending upon the sign of $y_* := h(x_*)$. (Clearly, the case $y_* = 0$ is trivial.)

Case 1: $y_* > 0$. Consider Figure 4.4. The new potential q_{k+1} is given by twice the area of the triangle $\triangle CDE$ times the slope of the line $\overline{BD}$. Since $\triangle CDE \subset \triangle ADE$, this means that

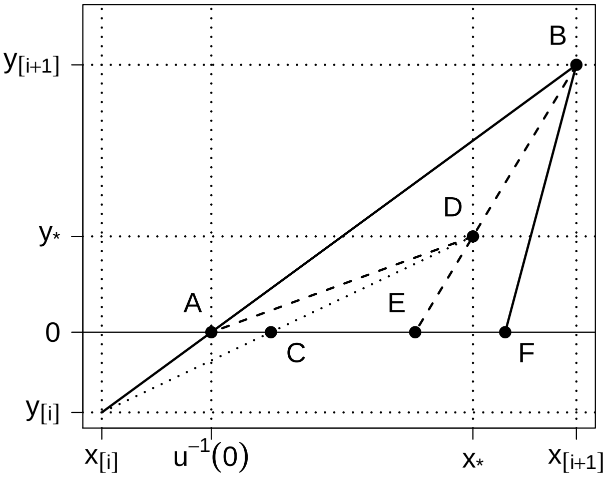


Figure 4.4: Proof of Lemma 4.2, Case 1: $y_* > 0$. The newly evaluated point is D. Here, slope($\overline{AB}$) $= \Delta$ and slope($\overline{BF}$) $= z^+_{[i+1]}$.

$$q_{k+1} \leq 2 \cdot \text{area}(\triangle ADE) \cdot \text{slope}(\overline{BD}) = y_* y_{[i+1]} \left(\Delta_k^{-1} \frac{y_{[i+1]} - y_*}{x_{[i+1]} - x_*} - 1 \right).$$

The above expression is maximized with respect to y_* at

$$y_* = \frac{1}{2}(y_{[i+1]} - \Delta_k(x_{[i+1]} - x_*)) = y_{[i+1]} \frac{\sqrt{z^+_{[i+1]}\Delta_k^{-1}} - 1}{2\sqrt{z^+_{[i+1]}\Delta_k^{-1}} - 1}.$$

Therefore

$$q_{k+1} \leq \left(y_{[i+1]}\right)^2 \frac{\left(\sqrt{z_{[i+1]}^+ \Delta_k^{-1}} - 1\right)^2}{2\sqrt{z_{[i+1]}^+ \Delta_k^{-1}} - 1},$$

which, in light of (4.2), proves (4.5).

Case 2: $y_* < 0$. Consider Figure 4.5. In this case, the new potential q_{k+1} is twice the area of $\triangle BCD$ times the slope of $\overline{BC}$. Since $\triangle BCD \subset \triangle ABC$,

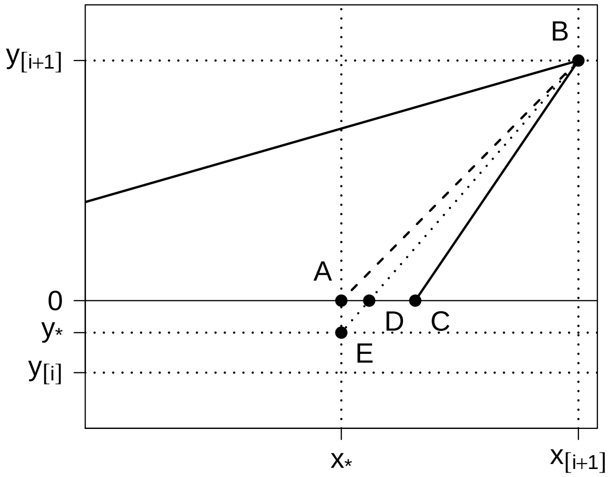


Figure 4.5: Proof of Lemma 4.2, Case 2: $y_* < 0$. The newly evaluated point is E. Here, slope$(\overline{BC}) = z_{[i+1]}^+$.

$$\begin{aligned} q_{k+1} &\leq 2 \cdot \text{area}(\triangle ABC) \cdot \text{slope}(\overline{BC}) \\ &= y_{[i+1]} \left(\left(x_{[i+1]} - x_*\right) z_{[i+1]}^+ - y_{[i+1]} \right) \\ &= \left(y_{[i+1]}\right)^2 \frac{\left(\sqrt{z_{[i+1]}^+ \Delta_k^{-1}} - 1\right)^2}{2\sqrt{z_{[i+1]}^+ \Delta_k^{-1}} - 1}, \end{aligned}$$

where the final equality follows from (4.4). Hence, (4.5) holds in this case as well. □

We now analyze the ratio of slopes $z_{[i+1]}\Delta_k^{-1}$, which appears in many of the above expressions. First we construct a sequence $(\gamma_k : k \geq 2)$ which, as we will see in Lemma 4.3 below, provides a lower bound on the sequence of slopes $(\Delta_k : k \geq 2)$. Refer to Figure 4.6 and define

$$\gamma_k = \text{slope}(\overline{AB}) \vee \text{slope}(\overline{AC}) = \left(\frac{-y_{[i]} z^+_{[i+1]}}{\left(x_{[i+1]} - x_{[i]}\right) z^+_{[i+1]} - y_{[i+1]}} \right) \vee z^-_{[i]}, \qquad (4.6)$$

where $\mathcal{B}_k = \left[x_{[i]}, x_{[i+1]}\right)$.

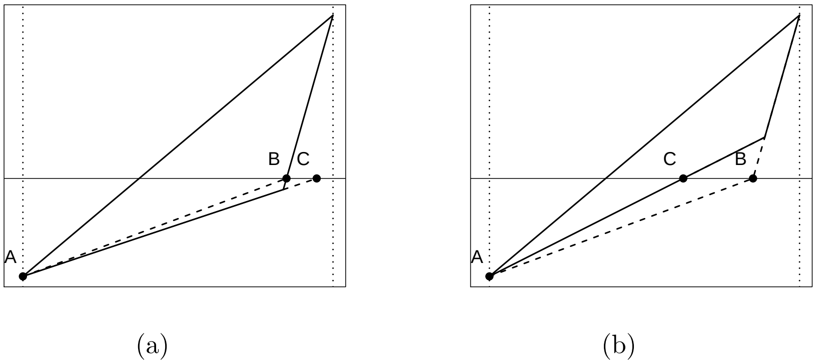


Figure 4.6: Construction of γ_k. See (4.6).

Although the sequence $(z^+_{[i+1]}\Delta_k^{-1} : k \geq 2)$ might not decrease in k, Lemma 4.3 shows that $(z^+_{[i+1]}\gamma_k^{-1} : k \geq 2)$ does. Moreover, the lemma also shows that the former sequence is bounded by the latter.

For the rest of this section, denote respectively by $x_{(l)}, y_{(l)}$, and $z_{(l)}$ the ordinate, abscissa, and subgradient upper bound associated with the right endpoint of the bracket after l iterations, for $l \geq 2$.

Lemma 4.3. *Suppose* $2 \leq j \leq k$. *Then:*

$$\Delta_k \geq \gamma_k \geq \gamma_j \qquad (4.7)$$

and

$$z_{(j)} \geq z_{(k)}. \qquad (4.8)$$

Hence,

$$z_{(k)}\Delta_k^{-1} \le z_{(j)}\gamma_j^{-1} \tag{4.9}$$

for all such j and k.

Proof. The only statement requiring proof is the second inequality in (4.7), as all the other inequalities follow from the construction of the relevant quantities. Consider Figure 4.7. The four panels show all four possible configurations after k iterations. The point E represents the sampled value at iteration $k+1$. In the left two panels, we have slope$(\overline{AB})$ > slope$(\overline{AC})$ and on the right the reverse is true; in the upper panels, the function value at the newly sampled point is positive and in the lower panels it is negative. In all cases, the new value γ_{k+1} is bounded below by the slope of $\overline{AF}$, which exceeds γ_k. □

We now state and prove the main result of this section.

Theorem 4.4. *Suppose $2 \le j \le k$. Let $\epsilon > 0$ be arbitrary. Define*

$$\eta_j := 2 \wedge \sqrt{z_{(j)}\gamma_j^{-1}}, \qquad \theta_j := \sqrt{\frac{2\eta_j^2 + \eta_j - 1}{\eta_j - 1}}.$$

If

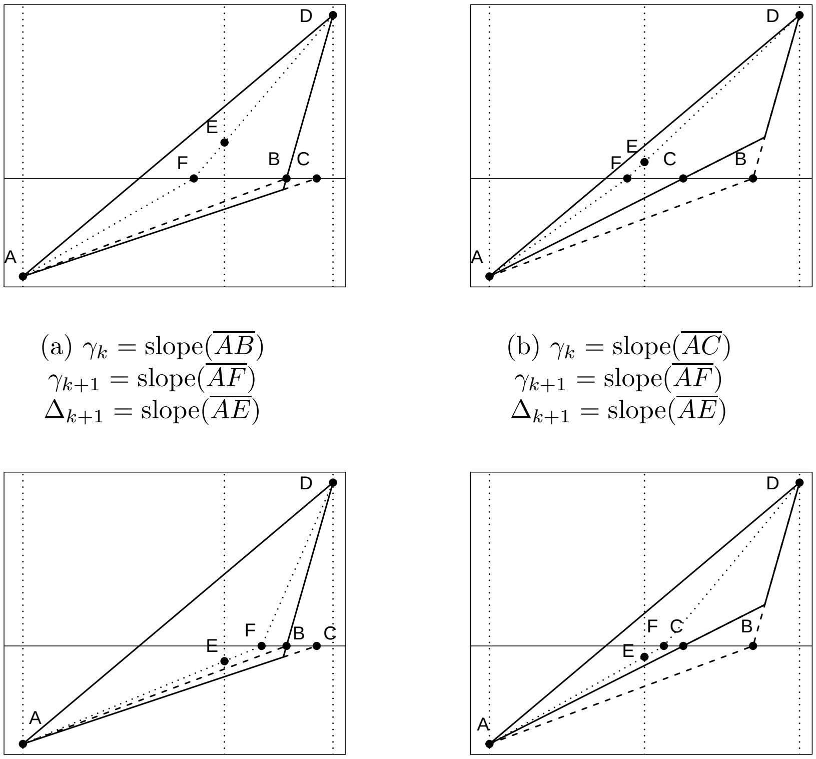
$$k \ge j + 1 + \frac{1}{2}\log_{\theta_j}\left[q_j \frac{\sqrt{z_{(j)}\Delta_j^{-1}} - 1}{2z_{(j)}\Delta_j^{-1} + \sqrt{z_{(j)}\Delta_j^{-1}} - 1}\right] + \log_{\theta_j}\frac{1}{2\epsilon}, \tag{4.10}$$

then

$$0 \le h\left(x_{(k)} - \frac{y_{(k)}}{z_{(k)}^{+}}\right) \le \epsilon$$

for all $h \in \mathcal{H}_k$.

Proof. One can check that as a function of η_j, the quantity θ_j decreases on $[1,2]$ and then increases on $[2,\infty)$. But by construction, $1 \le \eta_j \le 2$, and so Lemma 4.3



(a) $\gamma_k = \text{slope}(\overline{AB})$
$\gamma_{k+1} = \text{slope}(\overline{AF})$
$\Delta_{k+1} = \text{slope}(\overline{AE})$

(b) $\gamma_k = \text{slope}(\overline{AC})$
$\gamma_{k+1} = \text{slope}(\overline{AF})$
$\Delta_{k+1} = \text{slope}(\overline{AE})$

(c) $\gamma_k = \text{slope}(\overline{AB})$
$\gamma_{k+1} = \text{slope}(\overline{EF})$
$\Delta_{k+1} = \text{slope}(\overline{ED})$

(d) $\gamma_k = \text{slope}(\overline{AC})$
$\gamma_{k+1} = \text{slope}(\overline{EF})$
$\Delta_{k+1} = \text{slope}(\overline{ED})$

Figure 4.7: Configurations in proof of Lemma 4.3.

implies $\theta_j \leq \theta_{j+1} \leq \cdots \leq \theta_k$ and

$$\theta_l^2 \leq \frac{2z_{(l)}\Delta_l^{-1} + \sqrt{z_{(l)}\Delta_l^{-1}} - 1}{\sqrt{z_{(l)}\Delta_l^{-1}} - 1},$$

for $l =\geq 2$. Now, Lemma 4.2 and condition (4.10) imply

$$\theta_j^{2k-2(j+1)} \geq \frac{q_j}{4\epsilon^2} \cdot \frac{\sqrt{z_{(j)}\Delta_j^{-1}} - 1}{2z_{(j)}\Delta_j^{-1} + \sqrt{z_{(j)}\Delta_j^{-1}} - 1} \geq \frac{q_{j+1}}{4\epsilon^2}.$$

Therefore

$$q_{j+1} \leq 4\epsilon^2 \prod_{l=j+1}^{k-1} \theta_l^2 \leq 4\epsilon^2 \prod_{l=j+1}^{k-1} \frac{2z_{(l)}\Delta_l^{-1} + \sqrt{z_{(l)}\Delta_l^{-1}} - 1}{\sqrt{z_{(l)}\Delta_l^{-1}} - 1}.$$

So by Lemma 4.2,

$$q_k = \frac{q_k}{q_{k-1}} \frac{q_{k-1}}{q_{k-2}} \cdots \frac{q_{j+2}}{q_{j+1}} q_{j+1} \le 4\epsilon^2.$$

The result now follows from Lemma 4.1. □

The rate θ_j appearing in Theorem 4.4 is bounded below by 3. Therefore, Theorem 4.4 implies that rate of linear convergence to a δ-root is at least 3, as mentioned in the beginning of §4.2. We end this section with Algorithm 5, a formal statement of the deterministic convex δ-root finding algorithm.

Algorithm 5 Deterministic convex δ-root finding.

```
Require: Initial values $x_1 = a, x_2 = b, y_1 < 0, y_2 > 0$
  Set $z_1^- = z_2^- = 0, z_1^+ = z_2^+ = c, \Delta_2 = (y_2 - y_1)/b$
  Set $i = 1$, $k = 2$ ($i$ is bracket left endpoint, $k$ is iteration)
  while $u_k^{\leftarrow}(\delta) \le l_k^{\rightarrow}(-\delta)$ do
    Set $x_* = x_{[i+1]} - y_{[i+1]} \frac{\Delta_k^{-1}}{2\sqrt{z_{[i+1]}^+ \Delta_k^{-1}} - 1}$
    Set $y_* = h_*(x_*)$
    if $y_* \le 0$ then
      Set $i = i + 1$
    end if
    Set $y_{k+1} = y_*, x_{k+1} = x_*$
    Set $k = k + 1$
    Recompute the functions $u_k$ and $l_k$
  end while
  return $\frac{1}{2}\left(u_k^{\leftarrow}(\delta) + l_k^{\rightarrow}(-\delta)\right)$
```

4.3 Stochastic δ-root finding

We now turn to the stochastic setting, in which only interval estimates of $h_*(x)$ are available. In particular, we assume that we can generate *confidence intervals* for $h_*(x)$, i.e., random intervals (Y^-, Y^+) of a specified length which satisfy $Y^- \le h_*(x) \le Y^+$ with a specified probability. Such an interval can be constructed through a central limit theorem procedure, or through the use of Chebyshev- or Hoeffding-type inequalities as on pp. 21–22 of (Fishman 1996).

The bisection algorithm works in this case as well. Suppose we wish to report a δ-root at confidence level $1-\alpha$, for some $\alpha > 0$. At each x at which we sample h_*, we generate a confidence interval of width δ for $h_*(x)$ at a certain confidence level $1-\beta$. If this confidence interval contains 0, the sampled point is reported as a δ-root. Otherwise, the algorithm proceeds just as in the deterministic case. If every confidence interval actually contains the true value of h_* at the appropriate point, then the algorithm is guaranteed to return a δ-root. Since at most $\lceil \log_2((b-a)c/\delta)-1\rceil$ evaluations are needed, Bonferroni's inequality implies that the entire procedure succeeds with probability at least $1-\beta\lceil \log_2((b-a)c/\delta)-1\rceil$. Thus we take $\beta = \alpha/\lceil \log_2((b-a)c/\delta)-1\rceil$.

In the routine just described, it is sensible to take the confidence levels β to be the same across iterations. The reason for this is that it is known in advance that after each iteration, either the algorithm terminates or the bracket width is cut exactly in half. In contrast, an extension of our convex δ-root finding procedure of §4.2 allows us to select the confidence level for each iteration adaptively according to a bound on the work remaining.

In the remainder of this section, we describe how the convex δ-root finding procedure must be modified to accommodate interval estimates and give a probabilistic performance guarantee. We first analyze the algorithm under the assumption that the interval estimates are *guaranteed* to contain the true function value. We then relax this assumption and address the issue of how to choose the confidence levels for each interval estimate.

4.3.1 Envelope functions

In this section we describe how to compute the envelopes u_k and l_k in the stochastic setting. The proofs in this section are given as sketches only; complete proofs appear in the Appendix. Let $\mathcal{H}_0$ be defined as in §4.2. We define the *state* after k points have been sampled to be the triple $(\mathbf{x}_k, \mathbf{y}_k^-, \mathbf{y}_k^+)$, where $\mathbf{x}_k = (x_1, \ldots, x_k)$, $\mathbf{y}_k^- = (y_1^-, \ldots, y_k^-)$ and $\mathbf{y}_k^+ = (y_1^+, \ldots, y_k^+)$, and define

$$\mathcal{H}_k = \{h \in \mathcal{H}_0 : y_j^- \leq h(x_j) \leq y_j^+, j = 1, \ldots, k\}.$$

Hence, $\mathbf{y}_k^-$ and $\mathbf{y}_k^+$ represent lower and upper bounds on the true function values at the points in $\mathbf{x}_k$.

Let $\mathcal{H}_k(x)$ denote $\{h(x) : h \in \mathcal{H}_k\}$. Each time an interval estimate at a new point is generated, we wish to update the state in such a way that $y_j^- = \inf \mathcal{H}_k(x_j)$ and $y_j^+ = \sup \mathcal{H}_k(x_j)$ for all $j = 1, \ldots, k$. This may mean tightening some of the interval estimates previously generated; in particular, the interval (y_j^-, y_j^+) may be different from the confidence interval estimate originally generated for $h_*(x_j)$ at the j'th iteration. In order to distinguish between the original and the tightened estimates, we use capital letters (e.g., (Y_j^-, Y_j^+)) for the former.

We now construct the envelope functions l_k and u_k in such a way that for all $x \in [a, b]$, we have

$$\mathcal{H}_k(x) = [l_k(x), u_k(x)] \tag{4.11}$$

We say that the state $(\mathbf{x}_k, \mathbf{y}_k^-, \mathbf{y}_k^+)$ is *j-upper consistent* (respectively, *j-lower consistent*) if there exists a function $h \in \mathcal{H}_k$ such that $h(x_j) = y_j^+$ (respectively, $h(x_j) = y_j^-$), for $j = 1, \ldots, k$. If $(\mathbf{x}_k, \mathbf{y}_k^-, \mathbf{y}_k^+)$ is j-upper consistent (j-lower consistent) for all $j = 1, \ldots, k$, we say it is *upper consistent* (*lower consistent*). If the triple is both upper consistent and lower consistent, it is called *consistent*.

Given any state triple (not necessarily consistent), set $z^-_{[1]} = 0, z^+_{[k]} = c$, and

$$\begin{aligned} z^-_{[j]} &= 0 \vee \frac{y^-_{[j]} - y^+_{[j-1]}}{x_{[j]} - x_{[j-1]}} \vee \cdots \vee \frac{y^-_{[j]} - y^+_{[1]}}{x_{[j]} - x_{[1]}}, \text{ for } j = 2, \ldots, k, \text{ and} \\ z^+_{[j]} &= c \wedge \frac{y^+_{[j+1]} - y^-_{[j]}}{x_{[j+1]} - x_{[j]}} \wedge \cdots \wedge \frac{y^+_{[k]} - y^-_{[j]}}{x_{[k]} - x_{[j]}}, \text{ for } j = 1, \ldots, k-1. \end{aligned} \tag{4.12}$$

Now define the envelopes l_k and u_k by their actions on $x \in [x_{[j]}, x_{[j+1]}]$ as follows:

$$\begin{aligned} l_k(x) &= \left(y^-_{[j]} + \left(x - x_{[j]}\right) z^-_{[j]}\right) \vee \left(y^-_{[j+1]} - \left(x_{[j+1]} - x\right) z^+_{[j+1]}\right), \\ u_k(x) &= y^+_{[j]} + \frac{y^+_{[j+1]} - y^+_{[j]}}{x_{[j+1]} - x_{[j]}} \left(x - x_{[j]}\right), \end{aligned}$$

for $j = 1, \ldots, k-1$.

Proposition 4.5 shows that if $(\mathbf{x}_k, \mathbf{y}^-_k, \mathbf{y}^+_k)$ is appropriately consistent, then z^-_j and z^+_j provide tight bounds on the subgradient $\partial h(x_j)$ of every $h \in \mathcal{H}_k$ at x_j, given by

$$\partial h(x) = \{z : h(x') \geq h(x) + z(x' - x) \text{ for all } x' \in [a, b]\},$$

Proposition 4.6 shows that if $(\mathbf{x}_k, \mathbf{y}^-_k, \mathbf{y}^+_k)$ is consistent, then the envelopes provide tight bounds on the functions in $\mathcal{H}_k$.

prop 4.5. *Let $h \in \mathcal{H}_k$. For $j = 2, \ldots, k$ and $z \in \partial h(x_{[j]})$, we have $z \geq z^-_{[j]}$. For $j = 1, \ldots, k-1$ and $z \in \partial h(x_{[j]})$, we have $z \leq z^+_{[j]}$. Furthermore, for all $j = 1, \ldots, k$, if $(\mathbf{x}_k, \mathbf{y}^-_k, \mathbf{y}^+_k)$ is both upper consistent and j-lower consistent then there exists a function $\phi_j \in \mathcal{H}_k$ such that $\phi_j(x_{[j]}) = y^-_{[j]}$ and $z^-_{[j]}, z^+_{[j]} \in \partial \phi_j(x_{[j]})$.*

Sketch. The first two statements follow from the definitions of $z^-_{[j]}$ and $z^+_{[j]}$ as well as from elementary facts about convex functions. The function ϕ_j is constructed by splicing together the function u_k and the two lines passing through the point $(x_{[j]}, y^-_{[j]})$ with respective slopes $z^-_{[j]}$ and $z^+_{[j]}$. It is straightforward to do so such that $\phi_j \in \mathcal{H}_k$. □

prop 4.6. *If the state triple* $(\mathbf{x}_k, \mathbf{y}_k^-, \mathbf{y}_k^+)$ *is consistent then* (4.11) *holds.*

Sketch. By the convexity of $\mathcal{H}_k(x)$, we need only show that for $h \in \mathcal{H}$ and $x \in [a,b]$ we have $l_k(x) \le h(x) \le u_k(x)$ and that $l_k(x), u_k(x) \in \mathcal{H}_k(x)$.

For the upper bound, if $h \in \mathcal{H}_k$, then convexity of h implies $h(x) \le u_k(x)$ for all $x \in [a,b]$. The fact that $u_k(x) \in \mathcal{H}_k(x)$ for all such x is shown by taking $h_i, h_j, h_k \in \mathcal{H}_k$ which exhibit upper consistency at $x_{[i]}, x_{[j]}, x_{[l]}$, respectively, and using the known properties of these functions to prove $u_k \in \mathcal{H}_0$.

For the lower bound, again convexity of h implies $h(x) \ge l_k(x)$ for all $x \in [a,b]$. To prove that the lower bound is actually achieved, we construct a function ϕ by

$$\phi(x) = \begin{cases} \phi_j(x) & \text{if } a \le x \le x_{[j]} \\ l_k(x) & \text{if } x_{[j]} \le x \le x_{[j+1]} \\ \phi_{j+1}(x) & \text{if } x_{[j+1]} \le x \le b. \end{cases}$$

This function achieves the lower bound on $[x_j, x_{j+1}]$. □

In light of Proposition 4.6, we require a procedure which maintains the consistency of the state as h_* is evaluated at new points. We now present such a procedure. For notational ease, let $(\mathbf{x}_k, \mathbf{y}_k^-, \mathbf{y}_k^+) =: ((\tilde{x}_j), (\tilde{y}_j^-), (\tilde{y}_j^+))$ denote the values in the state triple *before* the $(k+1)^{\text{th}}$ point is evaluated and let $(\mathbf{x}_{k+1}, \mathbf{y}_{k+1}^-, \mathbf{y}_{k+1}^+) =: ((x_j), (y_j^-), (y_j^+))$ denote said values *after* the $(k+1)^{\text{th}}$ point is evaluated and the new state is made to be consistent. Let $[Y^-, Y^+]$ be the interval estimate for $h_*(x_{k+1})$. Algorithm 6 updates the upper bounds $y_1^+, \ldots, y_{k+1}^+$; Algorithm 7 updates the lower bounds $y_1^-, \ldots, y_{k+1}^-$ and also produces $z_1^-, \ldots, z_{k+1}^-, z_1^+, \ldots, z_{k+1}^+$. Algorithm 7 assumes that Algorithm 6 has already been run.

In what follows, we assume that the newly sampled point appears in the i'th position in the ordered list of the $k+1$ distinct x-coordinates, where $i = 2, \ldots, k$. We omit discussion of the cases $k = 0, 1$, as these are straightforward.

To update the upper bounds, Algorithm 6 first checks to make sure $Y^+ < u_k(x_{k+1})$, for otherwise Y^+ imparts no new information. Assuming this is true, the algorithm then sweeps leftward from the rightmost point $x_{[k+1]}$ to $x_{[i]}$, ensuring that the bounded growth and convexity conditions are satisfied. It then sweeps rightward from $x_{[1]}$ to $x_{[i-1]}$ checking monotonicity and convexity in the same manner.

Algorithm 6 Upper Bound Tightening.

```
Require: Consistent state after $k$ iterations; interval estimate $[Y^-, Y^+]$ for
    $h_*(x_{k+1})$; newly sampled point is $x_{k+1} = x_{[i]}$
    if $u_k(x_{[i]}) \leq Y^+$ then
        Set $y_j^+ = u_k(x_j)$ for all $j = 1, \ldots, k+1$
    else
        Set $y_{[i]}^+ = Y^+$
        Set $y_{[k+1]}^+ = \tilde{y}_{[k]}^+ \wedge \left(Y^+ + \left(x_{[k+1]} - x_{[i]}\right) c\right)$
        for $j = k$ to $i+1$ do
            Set $y_{[j]}^+ = \tilde{y}_{[j-1]}^+ \wedge \left(Y^+ + \left(x_{[j]} - x_{[i]}\right) \frac{y_{[j+1]}^+ - Y^+}{x_{[j+1]} - x_{[i]}}\right)$
        end for
        Set $y_{[1]}^+ = \tilde{y}_{[1]}^+ \wedge Y^+$
        for $j = 2$ to $i-1$ do
            Set $y_{[j]}^+ = \tilde{y}_{[j]} \wedge \left(Y^+ - \left(x_{[i]} - x_{[j]}\right) \frac{Y^+ - y_{[j-1]}^+}{x_{[i]} - x_{[j-1]}}\right)$
        end for
    end if
```

Algorithm 7 proceeds *outward* from $x_{[i]}$, alternating between updating the subgradient bounds and the function value lower bound at each point visited.

Proposition 4.7 says that Algorithms 6 and 7 maintain consistency of the state. More precisely, if the state is consistent after k iterations, then there are only two possibilities after the $(k+1)^{\text{th}}$ point is sampled:

1. the state remains consistent, or
2. the set $\mathcal{H}_{k+1}$ is empty.

Since we know $h_* \in \mathcal{H}$, possibility 2 can occur only if one of the interval estimates fails to contain the true function value at some stage.

Algorithm 7 Lower Bound Tightening.

```
Require: Algorithm 6 has been run
  Set $y^-_{[i]} = Y^- \vee l_k(x_{[i]})$
  Set $z^-_{[i]}$ and $z^+_{[i]}$ according to (4.12)
  for $j = i+1$ to $k+1$ do
    Set $y^-_{[j]} = \tilde{y}^-_{[j-1]} \vee \left( y^-_{[j-1]} + \left( x_{[j]} - x_{[j-1]} \right) z^-_{[j-1]} \right)$
    Set $z^-_{[j]}$ and $z^+_{[j]}$ according to (4.12)
  end for
  for $j = i-1$ to $1$ do
    Set $y^-_{[j]} = \tilde{y}^-_{[j]} \vee \left( y^-_{[j+1]} - \left( x_{[j+1]} - x_{[j]} \right) z^-_{[j+1]} \right)$
    Set $z^-_{[j]}$ and $z^+_{[j]}$ according to (4.12)
  end for
```

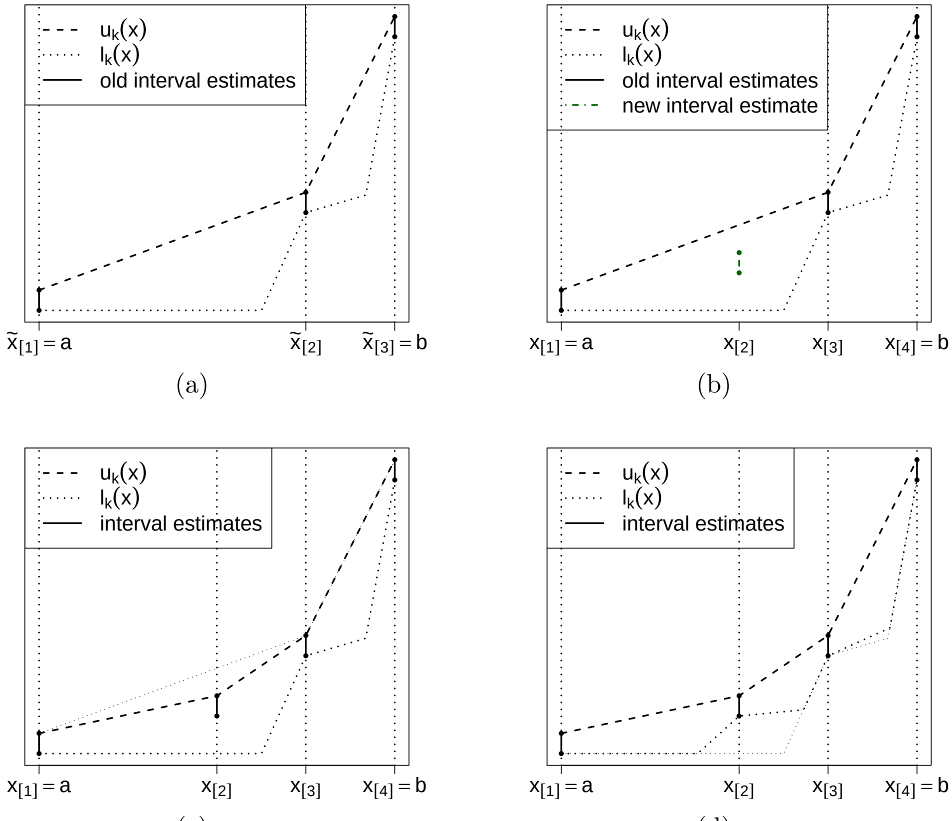
prop 4.7. *Suppose that after k points have been evaluated, the state $(\mathbf{x}_k, \mathbf{y}^-_k, \mathbf{y}^+_k)$ is consistent. (If $k = 1$, consistency merely requires that $\mathcal{H}_k$ is not empty.) Let $\mathcal{H}_{k+1}$ be the new triple $(\mathbf{x}_{k+1}, \mathbf{y}^-_{k+1}, \mathbf{y}^+_{k+1})$ produced by running first Algorithm 6, then Algorithm 7, on $(\mathbf{x}_k, \mathbf{y}^-_k, \mathbf{y}^+_k)$ and the new interval estimate $[Y^-, Y^+]$ of $h_*(x_{k+1})$. Then*

$$\mathcal{H}_{k+1} = \mathcal{H}_k \cap \{h : Y^- \leq h(x_{k+1}) \leq Y^+\}.$$

Moreover, either $(\mathbf{x}_{k+1}, \mathbf{y}^-_{k+1}, \mathbf{y}^+_{k+1})$ is consistent or $\mathcal{H}_{k+1}$ is empty.

Sketch. The inclusion $\mathcal{H}_{k+1} \subseteq \mathcal{H}_k \cap \{h : Y^- \leq h(x_{k+1}) \leq Y^+\}$ is immediate. Let $h \in \mathcal{H}_k$ satisfy $Y^- \leq h(x_{k+1}) \leq Y^+$. Then it can be shown by induction that after running Algorithm 6, $y_j^- \leq h(x_{[j]}) \leq y_j^+$ for all $j = 1, \ldots, k+1$. Upper consistency follows from $u_{k+1} \in \mathcal{H}_{k+1}$, which is easily shown. Finally, j-lower consistency is proved inductively on j by constructing a function ψ_j achieving the appropriate lower bounds. □

Figure 4.8 illustrates the process described in this section. Here, $k = 3$. Panel (a) depicts the state at the beginning of the $(k+1)^{\text{th}}$ iteration. Panel (b) depicts the new interval estimate. Panels (c) and (d) respectively depict the updating of the upper and lower envelopes.



(a)

(b)

(c)

(d)

Figure 4.8: Algorithms 6 and 7.

4.3.2 Reductions in potential

The development in this section mirrors that in §4.2.2. Indeed, many of the definitions and results in that section carry over to the case of interval estimates with minimal changes. The key assumption we make is that all interval estimates have width δ or less. As mentioned in the introduction to §4.3, after each function value is estimated the algorithm has enough information to terminate or to determine whether the root lies to the left or the right of the just-evaluated point. Throughout the remainder of this section, we assume $\mathcal{B}_k = [x_{[i]}, x_{[i+1]})$ and that an interval estimate of length less than or equal to δ has been generated for $h_*(x_{[i+1]})$. (This last assumption is critical for the next two lemmas.)

We define the potential in the current setting by

$$q_k = \left(y^+_{[i+1]}\right)^2 \left(z^+_{[i+1]}\Delta_k^{-1} - 1\right), \tag{4.13}$$

where

$$\Delta_k = \frac{y^+_{[i+1]} - y^+_{[i]}}{x_{[i+1]} - x_{[i]}}.$$

Compare with (4.2). Lemmas 4.8 and 4.9 are analogous to Lemmas 4.1 and 4.2. The proofs are virtually identical to those in §4.2, and so we omit them here.

Lemma 4.8. *Let $\hat{x} = x_{[i+1]} - y^+_{[i+1]}/z^+_{[i+1]}$. Then*

$$-\delta \leq l_k\left(\hat{x}\right) \leq u_k\left(\hat{x}\right) \leq \frac{1}{2}\sqrt{q_k}.$$

In particular, if $q_k \leq 4\delta^2$, then $\hat{x}$ is a δ-root.

Lemma 4.9. *If the $(k+1)^{th}$ sampling point is given by (4.4), then*

$$\frac{q_{k+1}}{q_k} \leq \left(\frac{y^+_{[i+1]} + \delta}{y_{[i+1]}}\right)^2 \frac{\sqrt{z^+_{[i+1]}\Delta_k^{-1}} - 1}{2z^+_{[i+1]}\Delta_k^{-1} + \sqrt{z^+_{[i+1]}\Delta_k^{-1}} - 1}. \tag{4.14}$$

The way in which Lemmas 4.1 and 4.8 differ is not significant, since in both cases a potential of less than $4\delta^2$ indicates that the algorithm may terminate. On the other hand, the difference between Lemmas 4.2 and 4.9 matters. The present lemma contains a "slippage" factor of $((y^+_{[i+1]} + \delta)/y^+_{[i+1]})^2$. The good news is that when the potential is great, this factor is unimportant. It only significantly slows the guaranteed reduction in potential during the last three iterations of the algorithm.

We now derive an upper bound on the potential reduction given by (4.14). The key idea is to reparameterize the right-hand side of (4.14) in a way which allows us to maximize the bound. Let us take

$$\rho_j := \sqrt{z^+_{[i+1]}\Delta_k^{-1}} \quad \text{and} \quad w_k := \frac{\sqrt{q_k}}{\delta} = \frac{y^+_{[i+1]}}{\delta}\sqrt{\rho_k^2 - 1}.$$

Take $\hat{x}$ as in Lemma 4.8. If the algorithm has not yet terminated after k steps, then we must have

$$\delta \leq u(\hat{x}) = y^+_{[i+1]} - (x_{[i+1]} - \hat{x})\Delta_k = y^+_{[i+1]}\frac{\rho_k^2 - 1}{\rho_k^2}.$$

Hence, $w_k \geq \rho_k^2/\sqrt{\rho_k^2 - 1}$. This implies $\rho_k \in R(w_k)$, where

$$R(w) = \left[\frac{1}{2}\left(\sqrt{w(w+2)} - \sqrt{w(w-2)}\right), \frac{1}{2}\left(\sqrt{w(w+2)} + \sqrt{w(w-2)}\right)\right].$$

Here, we have used the quadratic formula and the fact that:

$$\frac{1}{2}(\sqrt{w+2} + \sqrt{w-2})^2 = w + \sqrt{w^2 - 4}.$$

So the next-step potential reduction on the right-hand side of (4.14) is bounded above by $\sigma(w_k)$, where

$$\sigma(w) := \sup_{\rho \in R(w)} \left(1 + \frac{\sqrt{\rho^2 - 1}}{w}\right)^2 \frac{\rho - 1}{2\rho^2 + \rho - 1}.$$

Figure 4.9 contains plots of $\sigma(w)$ and of $w \mapsto w^2\sigma(w)$, both of which are easily computed numerically. Panel (a) demonstrates that $\sigma(w)$ is both decreasing in w and fairly close to $1/9$ so long as w is sufficiently great – at least about 40. Panel (b) suggess that $w^2\sigma(w)$ is monotone in w. This is an important point, for it allows us to conclude that a smaller value of the potential q_k always indicates less remaining work.

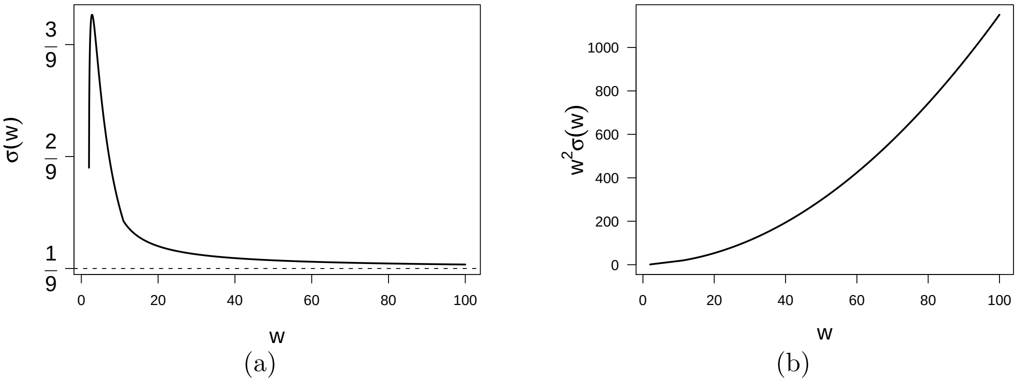


Figure 4.9: Plots of $\sigma(w)$ and $w^2\sigma(w)$.

We are now in a position to derive a bound on the number of remaining iterations given potential q_k. Let $v_0 = 2$, and for $s \geq 1$ define v_s by

$$v_s^2\sigma(v_s) = v_{s-1}^2. \tag{4.15}$$

Theorem 4.10 shows how the sequence (v_s) can be used to determine such a bound.

Theorem 4.10. *If $w_j^2 \leq v_s^2$ after j iterations then the algorithm terminates in at most s additional iterations.*

Proof. Follows from Lemma 4.8, Lemma 4.9 and the monotonicity of the map $w \mapsto w^2\sigma(w)$. □

By including the value of the slippage factor, we can tighten this somewhat.

Corollary 4.11. *If after j iterations,*

$$w_j^2 \left(\frac{y_{(j)} + \delta}{y_{(j)}} \right)^2 \frac{\rho_j - 1}{2\rho_j^2 + \rho_j - 1} \leq v_s^2,$$

then the algorithm terminates in at most $s+1$ additional iterations.

Proof. Immediate from the theorem and Lemma 4.9. □

The first few values of the sequence $(v_s : s \geq 0)$ are presented in Table 4.1. For

Table 4.1: The sequence $(v_s : s \geq 0)$.

v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
2	3.3769	7.1203	19.448	56.557	167.92	502.02	1504.3	4511.3	13532

$s > 3$, the ratio v_s/v_{s-1} is fairly close to 3, the guaranteed linear convergence rate in the deterministic case. For $s > 9$, this ratio is virtually indistinguishable from 3, and may be treated as equal to 3 for the purposes of computing a bound on the number of remaining iterations.

4.3.3 Selecting the confidence levels

We now relax the assumption that every interval estimate is guaranteed to enclose the true function value. Indeed, in the context of stochastic root finding, the best that one can hope for is that a given interval covers the true function value with high probability. By carefully choosing the confidence levels of such intervals, we can construct a probabilistic guarantee for our root-finding procedure. More precisely, we can promise that for a given $\alpha \in (0, 1)$, the probability that the (random) point our algorithm returns is indeed a δ-root is at least $1 - \alpha$. This statement is similar in principle to the probability-of-correct-selection guarantees that come with ranking and selection procedures; see (Kim and Nelson 2006).

The key assumptions we make are that

1. conditional on the (random) abscissa x_j at which an interval is generated, the interval $[Y_j^-, Y_j^+]$ is independent of all previous steps in the algorithm, and
2. conditional on x_j, the interval $[Y_j^-, Y_j^+]$ contains the true function value $h_*(x_j)$ with probability at least $1 - \beta_j$, where we can select β_j.

The bound derived in the previous section on the number of remaining iterations guides our selection of β_j.

Recall that the initial information available about the unknown non-decreasing convex function h_* consists of the domain $[a, b]$ of h_* and an upper bound c on the subgradient of h_* on (a, b). In order for the analysis of the previous section to apply, we must have the root bracketed at each step; in particular, in order for Lemmas 4.8 and 4.9 to hold, the interval estimate at the right endpoint of the bracket must be to within δ. Thus, the algorithm begins by generating a length-δ interval estimate of $h_*(b)$. It is not necessary to have the left endpoint estimated to that degree of accuracy. Indeed, we may take the initial interval estimate of the left endpoint to be the *a priori* bounds $(Y_1^-, Y_1^+) = (-(b-a)c, 0)$. Observe that we may consider this estimate to be at confidence level 1, since there is no randomness.

Although we do not know what the resulting potential q_2 will be once $h_*(b)$ is estimated, we do know that

$$q_2 = \left(y^+_{[2]}\right)^2 \left(c\frac{b-a}{y^+_{[2]} - y^+_{[2]}} - 1\right) \leq (b-a)cy^+_{[2]} - \left(y^+_{[2]}\right)^2 \leq \frac{(b-a)^2c^2}{4} =: q^*_2.$$

Let $k^*(q)$ be the bound on the number of *additional* random intervals required before we can halt the algorithm, as provided by Theorem 4.10, when the potential is q. Let B_j be the event that at least j intervals are generated (counting the first two) in the algorithm, and let A_j be the union of B^c_j and the subset of B_j in which the jth interval contains the true function value ($j \geq 1$). In words, A_j represents the event that *either* fewer than j intervals are required, *or* j intervals or more are required, and the jth interval contains the true function value. Define the (trivial) values $Y^-_j = Y^+_j = 0$ on the event B^c_j, i.e., when the algorithm uses fewer than j intervals. Let $\mathcal{F}_j$ be the sigma field generated by $Y^-_1, Y^+_1, \ldots, Y^-_j, Y^+_j$, for $j = 1, 2, \ldots$, and notice that B_j is a member of $\mathcal{F}_{j-1}$ for all $j \geq 1$, with $\mathcal{F}_0$ being the trivial sigma field.

Now, $P(A_j) = EP(A_j|\mathcal{F}_{j-1})$. On the event B^c_j (so that $j-1$ or fewer intervals are required), $P(A_j|\mathcal{F}_{j-1}) = 1$. Furthermore, on the event B_j, this probability is equal to $1-\beta_j$ (it is here that we use the independence assumption). Hence, $P(A^c_j|\mathcal{F}_{j-1}) \leq \beta_j I(B_j)$, and so $P(A^c_j) \leq \beta_j$. Set $K_0 = 1 + k^*(q^*_2)$, which is a bound on the total number of intervals the algorithm must generate, and set $\beta_1 = 0$ and

$$\beta_j = \frac{\alpha}{K_0}$$

for all $j = \geq 1$. Then the probability that the reported root is indeed a δ-root is bounded below by

$$\begin{aligned}
P\left(\bigcap_{j=1}^{K_0} A_j\right) &\geq 1 - \sum_{j=1}^{K_0} P(A_j^c) \\
&\geq 1 - \sum_{j=1}^{K_0} \frac{\alpha}{K_0} \\
&= 1 - \alpha,
\end{aligned}$$

where we have used Bonferroni's inequality in the first step.

This somewhat brute-force approach to designing the overall root-finding procedure ignores the information contained in the intervals that are successively obtained. In particular, it requires that each interval have a uniformly high probability that it covers the true value, and since the intervals need to have width at most δ, large simulation run-lengths may be necessary at each x value. We now introduce a concept we call *adaptive α-spending* that still ensures that the algorithm has an α-guarantee, but will, in many cases, reduce the simulation run-lengths required to obtain the intervals. Algorithm 8 implements this idea.

The key idea is that, at each step of the algorithm, we compute a new bound on the number of steps that remain, and adjust the coverage probabilities of the remaining intervals accordingly. In other words, the sequence (β_j) of allowed probabilities of error may (and probably will) increase with j.

To see why Algorithm 8 has a guaranteed probability of success of at least $1-\alpha$ and is more efficient than the brute-force algorithm, redefine the events A_j and B_j to apply to Algorithm 8 rather than the "brute force" one. As above, we have that for $j \geq 1$, $P(A_j^c | \mathcal{F}_{j-1}) \leq \beta_j I(B_j)$. Let T be the value of j when the algorithm terminates, so that T is the number of intervals generated, including the initial two that are given. Notice that the sequence $K_0, K_1, \ldots, K_T$ is decreasing, so a

Algorithm 8 Stochastic δ-root finding via adaptive α-spending.

```
Require: Constants $\delta, c > 0$, $\alpha \in (0,1)$, and $a < b$
  Set $K_0 = 1 + k^*(q_2^*)$, $K_1 = K_0 - 1$, $\beta_1 = 0$, $\beta_2 = \alpha/K_0$
  Set $\mathbf{x}_2 = (a,b)$, $\mathbf{y}_2^- = (-(b-a)c, 0)$, $\mathbf{y}_2^+ = (0, (b-a)c)$
  Compute estimate $[Y_2^-, Y_2^+]$ for $h_*(x_2)$ at confidence level $1 - \beta_2$
  Update state with the estimate $(Y_2^-, Y_2^+)$ according to Algorithms 6 and 7
  Set $j = 2$, $K_j = k^*(q_j)$
  while $l_j^{\rightarrow}(-\delta) \geq u_j^{\leftarrow}(\delta)$ do
    Set $j = j + 1$
    Set $\beta_j = \frac{\alpha - (\beta_1 + \cdots + \beta_{j-1})}{K_{j-1}}$
    Select new point to sample according to (4.4)
    Generate interval estimate $(Y_j^-, Y_j^+)$ at confidence level $1 - \beta_j$
    Update state according to Algorithms 6 and 7
    Compute $q_j$ and $K_j = k^*(q_j)$ from state
  end while
  Return midpoint of all known $\delta$-roots
```

straightforward induction argument shows that for $j = 1, \ldots, T$, $\beta_1 + \cdots + \beta_j \leq \alpha$, and $\beta_j \geq \alpha/K_0$. The latter inequality shows that the new version of the algorithm is more efficient, as the individual confidence levels are no greater here than in the brute force version. Moreover, we then have that the probability that the algorithm returns a true δ-root is at least

$$\begin{aligned} P\left(\bigcap_{j=1}^{K_0} A_j\right) &\geq 1 - \sum_{j=1}^{K_0} P(A_j^c) \\ &= 1 - E\sum_{j=1}^{K_0} P(A_j^c|\mathcal{F}_{j-1}) \\ &\geq 1 - E\sum_{j=1}^{K_0} \beta_j I(B_j) \\ &= 1 - E\sum_{j=1}^{T} \beta_j \\ &\geq 1 - \alpha. \end{aligned}$$

4.3.4 Intermediate values of δ

The procedure we have described has the property that every confidence interval computed is of width δ. Typically, generating such a confidence interval requires an amount of work proportional to $1/\delta^2$. Since points sampled early in the algorithm may be far from the root, we may be able to save computational effort by initially solving the problem for a greater value of δ than the one we actually want.

We now discuss a simple example of how this idea might be implemented while maintaining the statistical guarantee of the original algorithm. We do not claim that the specifics provided here are in any way optimal. The approach we suggest is to solve the root-finding problem in its entirety for one or more intermediate values of the tolerance parameter δ. Here, we give details for a version which uses a single intermediate value.

Even before any function values are sampled, it is known that $-(1/2)(b-a)c \leq h_*((1/2)(a+b)) \leq (1/2)(b-a)c$, a fact which follows from the definition of $\mathcal{H}_0$. Hence, the initial problem is already solved for a tolerance parameter value of $(1/2)(b-a)c$. Choose some $\tilde{\delta}$ satisfying $\delta < \tilde{\delta} < (1/2)(b-a)c$. If we were to generate an interval estimate of $h_*(b)$ with width $\tilde{\delta}$ and having upper endpoint Y^+, the resulting value of the potential would be bounded above by $(Y^+)^2(c(b-a)/Y^+ - 1) \leq (1/4)c^2(b-a)^2$. Plug this bound and the value $\tilde{\delta}$ into Theorem 4.10 to compute $\tilde{k}_{\max}$, the maximum number of additional iterations required to find a $\tilde{\delta}$-root. Now compute the interval estimate of $h_*(b)$ with width $\tilde{\delta}$ and confidence level $1-\alpha/2(\tilde{k}_{\max}+1)$. Allocate error probability $\alpha/2 \times (1-1/(\tilde{k}_{\max}+1))$ to this subproblem and solve using Algorithm 8.

Once the subproblem is solved, after, say, $\tilde{k}$ steps, there are two cases to consider (assuming, of course, we did not happen to find a δ-root in solving the subproblem). The simpler case is that the interval estimate at the last point sampled (call it x_*) does not contain 0. In this case, the true root is bracketed between two adjacent points in $\mathcal{X}_{\tilde{k}}$. In fact, it is bracketed in a smaller interval, $[u^{\leftarrow}_{\tilde{k}}(\delta), l^{\rightarrow}_{\tilde{k}}(-\delta)]$. In a manner similar to how we dealt with the initial subproblem, we may compute an upper bound on the value of the potential after sampling h_* at the new bracket right endpoint with interval width δ, use Theorem 4.10 to determine the maximum number of iterations it will take to finish solving the problem, and spend the remaining error probability $\alpha/2$ accordingly.

The more difficult case is when the last point sampled is a $\tilde{\delta}$-root. In this case, we no longer have the true root bracketed. Hence, we must resample at x_* with interval width δ to bracket the root. This involves computing the maximum possible value of the potential after resampling at x_* *and possibly after resampling at the new bracket right endpoint* and allocating error probability in the usual way.

The procedure described above may, of course, be extended to include multiple intermediate tolerance parameters. As a heuristic, we suggest that the sequence of δ's decline geometrically to the target δ at rate 3.

We conclude this section with a worked example of the intermediate δ procedure in practice. We assume that the error with which h_* is sampled is standard normal. Suppose that the function h_* is given by

$$h_*(x) = (1/200)x^2 - (8/3).$$

Take $a = 0, b = 100, c = 1, \alpha = 0.05, \delta = 0.1$. The potential after sampling at b is bounded above by $(b-a)^2c^2/4 = 2,500$. Take $\tilde{\delta} = 3\delta = 0.3$. Using Theorem 4.10, we determine that a $\tilde{\delta}$-root will be found after at most 5 iterations of the algorithm. Thus, we begin with a confidence interval for $h_*(100)$ having width 0.3

and confidence level $1 - \alpha/(2 \times (1+5)) \approx .99583$.

We now invoke Algorithm 8, from which we request a $\tilde{\delta}$ root at confidence level $1 - (\alpha/2 - \alpha/12) \approx .97917$. Table 4.2 gives the sequence of points evaluated, the resulting interval estimates, the alpha spent at each iteration, the potential after each iteration, and the number of simulation trials used for the confidence interval estimate. Iteration 0 refers to the initial interval estimate of $h_*(100)$. Figure 4.10 plots the envelopes at all iterations of the subproblem. Table 4.3 contains a summary of the state after the subproblem is solved.

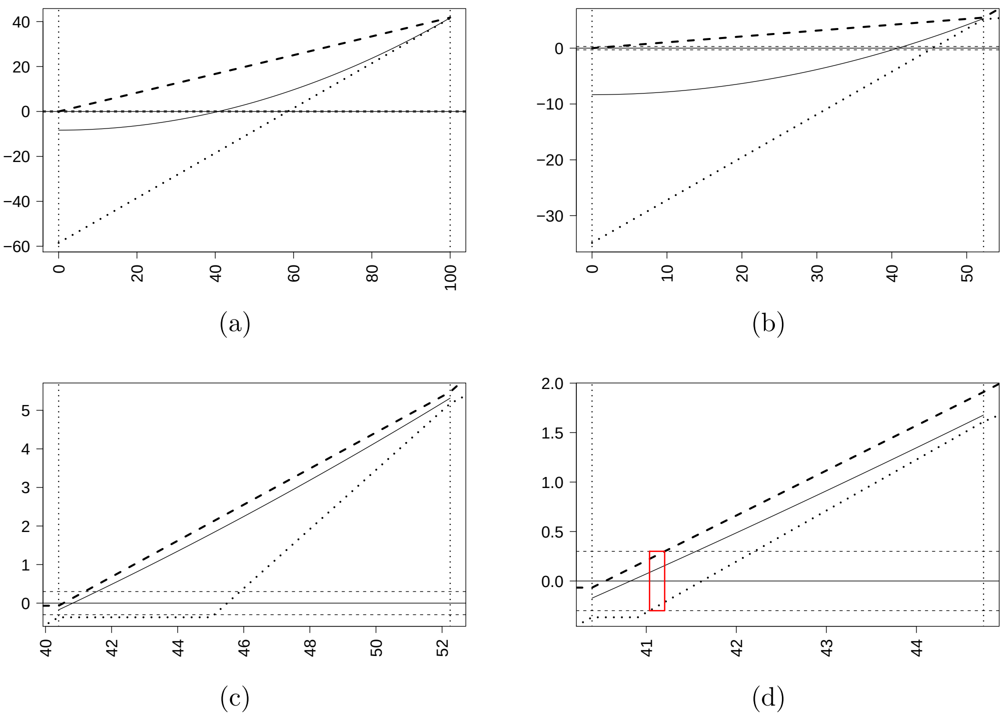
Table 4.2: Subproblem to find $\tilde{\delta}$-root, where $\tilde{\delta} = 0.3$.

iter.	**eval. pt.**	**int. est.**	**alpha spent**	**potential**	**# trials**
0	100	[41.48, 41.78]	.00417	2432.5	368
1	52.25	[5.17, 5.47]	.00417	189.28	368
2	40.40	[-0.37, -0.07]	.00417	19.152	368
3	44.75	[1.61, 1.91]	.00417	0.4837	368

Table 4.3: State after solving $\tilde{\delta}$-root subproblem.

	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$x_{[j]}$	0.0000	40.398	44.745	52.245	100.0000
$y^-_{[j]}$	-20.284	-0.36690	1.6105	5.1741	41.484
$y^+_{[j]}$	-0.06690	-0.06690	1.9105	5.4741	41.784
$z^-_{[j]}$	0.0000	0.0000	0.38586	0.44238	0.75406
$z^+_{[j]}$	0.49302	0.49302	0.51514	0.76662	1.0000

Once the $\tilde{\delta}$-root subproblem is solved, we proceed to solve the problem for the original tolerance parameter δ. Since the interval estimate at the last point sampled did not contain 0, we resample next at the point $x_* = l_3^{\rightarrow}(-\delta) = 41.42$. In order to figure out at what confidence level this interval should be estimated,



(a) (b) (c) (d)

Figure 4.10: Subproblem to find $\tilde{\tilde{\delta}}$-root, where $\tilde{\tilde{\delta}} = 0.3$.

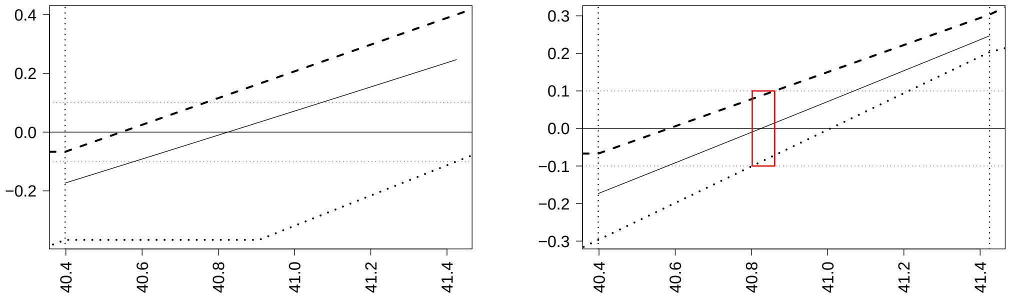
we must first bound the potential of the resulting state. Such a bound is given by

$$\begin{aligned}
&\sup_{\delta \le y \le u(x_*)} y^2 \left(\left[z^+_{[3]} \wedge \frac{y^+_{[3]} - (y - \delta)}{x_{[3]} - x_*} \right] \times \frac{x_* - x_{[2]}}{y - y^+_{[2]}} - 1 \right) \\
&= \left[\sup_{.1 \le y \le .3} y^2 \left(\frac{0.52894}{y + 0.06690} - 1 \right) \right] \vee \left[\sup_{.3 \le y \le .40015} y^2 \left(0.30923 \times \frac{2.0105 - y}{y + 0.06690} - 1 \right) \right] \\
&= .25290^2 \left(\frac{0.52894}{0.31980} - 1 \right) = 0.04183.
\end{aligned}$$

Hence, the value w of Theorem 4.10 is bounded above by $\sqrt{.04183}/\delta = 2.0452$, which implies that at most one additional iteration is needed after computing an interval estimate of $h_*(x_*)$. Thus, we use confidence level $1 - \alpha/4$ for the next interval estimate. It turns out that after simulating at x_*, condition (4.1) is met and the algorithm terminates. The final interval estimate and the plot of the state upon termination are, respectively, in Table 4.4 and Figure 4.11.

Table 4.4: Subproblem to find δ-root.

iter.	**eval. pt.**	**int. est.**	**alpha spent**	**potential**	**# trials**
0	41.42	[0.2033, 0.3033]	.0125	.0323	2496



(a)

(b)

Figure 4.11: Subproblem to find δ-root.

Finally, we note that the two-phase procedure used a total of 5 iterations and 3,968 simulation trials. Simply using Algorithm 8 at level δ, in contrast, required 5 iterations and a total of 13,676 simulation trials. It is likely that in practice the version of the procedure which considers a decreasing sequence of tolerance parameters will typically outperform the fixed-δ version as in this example.

Appendix A

Additional Proofs

We provide proofs of Propositions 4.5, 4.6, and 4.7. Collectively, these propositions demonstrate that at each iteration of the root-finding algorithm the envelope functions, as defined in §3.1 of Chapter 2, precisely define the set of functions consistent with the simulated confidence intervals.

Lemma 1. The state $(\mathbf{x}_k, \mathbf{y}_k^-, \mathbf{y}_k^+)$ is upper consistent if and only if $u_k \in \mathcal{H}_k$.

Proof. The fact that $u_k \in \mathcal{H}_k$ implies upper consistency is immediate. Conversely, suppose $\mathcal{H}_k$ is upper consistent. Accordingly, for $1 \leq i < j \leq k$, let $h_i, h_j \in \mathcal{H}_k$ satistfy $h_i(x_{[i]}) = y_{[i]}^+$ and $h_i(x_{[j]}) = y_{[j]}^+$. Then

$$u_k(x_{[i]}) = y_{[i]}^+ = h_i(x_{[i]}) \leq h_i(x_{[j]}) \leq y_{[j]}^+ = u_k(x_{[j]})$$

and

$$\begin{aligned} u_k(x_{[j]}) &= y_{[j]}^+ \\ &= h_j(x_{[j]}) \\ &\leq h_j(x_{[i]}) + (x_{[j]} - x_{[i]})c \\ &\leq y_{[i]}^+ + (x_{[j]} - x_{[i]})c \\ &= u_k(x_{[i]}) + (x_{[j]} - x_{[i]})c. \end{aligned}$$

Since u_k is the linear interpolant of $(x_{[1]}, y_{[1]}^+), \ldots, (x_{[k]}, y_{[k]}^+)$, this proves that u_k is nondecreasing and satisfies the bounded slope condition. Finally, if $1 \leq i < j <$

$l \leq k$, then

$$\begin{aligned}(x_{[l]} - x_{[i]})u_k(x_{[j]}) &= (x_{[l]} - x_{[i]})h_j(x_{[j]}) \\ &\leq (x_{[j]} - x_{[i]})h_j(x_{[l]}) + (x_{[l]} - x_{[j]})h_j(x_{[i]}) \\ &\leq (x_{[j]} - x_{[i]})y^+_{[l]} + (x_{[l]} - x_{[j]})y^+_{[i]} \\ &= (x_{[j]} - x_{[i]})u_k(x_{[l]}) + (x_{[l]} - x_{[j]})u_k(x_{[i]}),\end{aligned}$$

which proves that u_k is convex. Hence, $u_k \in \mathcal{H}_k$.

Proof of Proposition 4.5

Step One. Let $j = 2, \ldots, k$, $h \in \mathcal{H}_k$, and $z \in \partial h(x_j)$ be arbitrary. Since h is non-decreasing, $z \geq 0$. Furthermore, for any $l = 1, \ldots, j-1$,

$$z \geq \frac{h(x_{[j]}) - h(x_{[l]})}{x_{[j]} - x_{[l]}} \geq \frac{y^-_{[j]} - y^+_{[l]}}{x_{[j]} - x_{[l]}}.$$

Therefore,

$$z \geq 0 \vee \frac{y_{[j]} - y_{[j-1]}}{x_{[j]} - x_{[j-1]}} \vee \cdots \vee \frac{y^-_{[j]} - y^+_{[1]}}{x_{[j]} - x_{[1]}} = z^-_{[j]}.$$

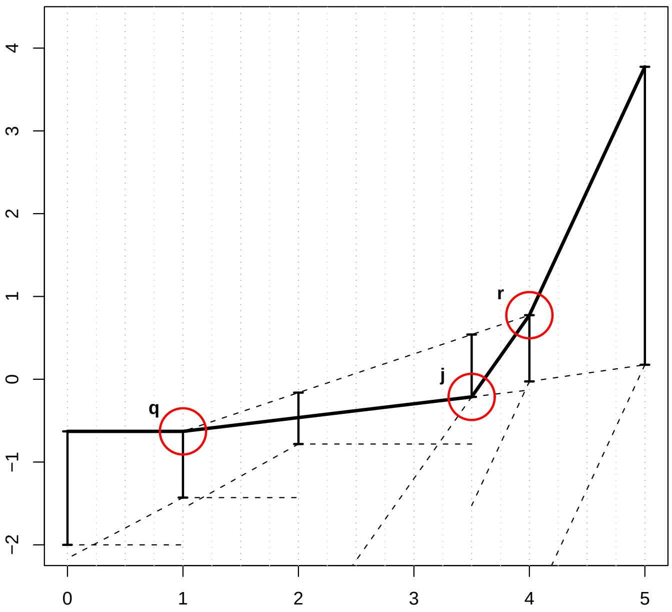
The statement $z \leq z^+_{[j]}$ for $j = 1, \ldots, k-1$ is proved similarly.

Step Two. Now take $j = 1, \ldots, k$ and suppose the state is both upper consistent and j-lower consistent. Let q be the least element of $\{1, \ldots, j-1\}$ such that $z^-_{[j]} = (y^-_{[j]} - y^+_{[q]})/(x_{[j]} - x_{[q]})$, or $q = 1$ if there is no such element (which could be the case if $z^-_{[j]} = 0$). Let r be the greatest element of $\{j+1, \ldots, k\}$ such that $z^+_{[j]} = (y^+_{[r]} - y^-_{[j]})/(x_{[r]} - x_{[j]})$, or $r = k$ if there is no such r. Now define ϕ_j by

$$\phi_j(x) = \begin{cases} y^-_{[j]}(x_{[j]} - x)z^-_{[j]} & \text{if } x_{[q]} \leq x \leq x_{[j]}, \\ y^+_{[j]}(x_{[j]} - x)z^-_{[j]} & \text{if } x_{[j]} \leq x \leq x_{[r]}, \\ u_k(x) & \text{otherwise;} \end{cases}$$

see Figure A.1. Now observe that if $q > 0$, then

Figure A.1: The function ϕ_j.



$$z^-_{[j]} = \frac{y^-_{[j]} - y^+_{[q]}}{x_{[j]} - x_{[q]}} > \frac{y^-_{[j]} - y^+_{[q-1]}}{x_{[j]} - x_{[q-1]}}.$$

Rearranging this gives

$$(y^-_{[j]} - y^+_{[q]})(x_{[j]} - x_{[q-1]} + x_{[q]} - x_{[q]}) > (y^-_{[j]} - y^+_{[q-1]} + y^+_{[q]} - y^+_{[q]})(x_{[j]} - x_{[q]}).$$

If we subtract $(y^-_{[j]} - y^+_{[q]})(x_{[j]} - x_{[q]})$ from both sides, we get

$$(y^-_{[j]} - y^+_{[q]})(x_{[q]} - x_{[q-1]}) > (y^+_{[q]} - y^+_{[q-1]})(x_{[j]} - x_{[q]}).$$

or,

$$z^-_{[j]} = \frac{y^-_{[j]} - y^+_{[q]}}{x_{[j]} - x_{[q]}} > \frac{y^+_{[q]} - y^+_{[q-1]}}{x_{[q]} - x_{[q-1]}} = \frac{\phi_j(x_{[q]}) - \phi_j(x_{[q-1]})}{x_{[q]} - x_{[q-1]}}. \tag{A.1}$$

Similarly, one can show that for $r < k$,

$$\frac{\phi_j(x_{[r+1]}) - \phi_j(x_{[r]})}{x_{[r+1]} - x_{[r]}} > z^+_{[j]}. \tag{A.2}$$

In words, these inequalities show that the slope of $\phi_j(x)$ increases as x passes over the boundaries of the separate regions on which ϕ_j is defined.

It is easy to check that ϕ_j is piecewise linear and continuous. Continuity, along with $z^-_{[j]} \geq 0$, the conclusion of Step One, and Lemma 1, imply that ϕ_j is nondecreasing. Observe that by invoking Lemma 1 we have used upper consistency. Similarly, we conclude that ϕ_j satisfies the bounded slope condition. Moreover, the aforementioned facts, combined with (A.1) and (A.2), imply that ϕ_j is convex. So we have established $\phi_j \in \mathcal{H}_0$.

In order to prove that in fact $\phi_j \in \mathcal{H}_k$, we must show

$$y^-_{[l]} \leq \phi_j \left(x_{[l]}\right) \leq y^+_{[l]}, \tag{A.3}$$

for $l = 1, \ldots, k$. Evidently, (A.3) holds for $1 \leq l < q$ or $r < l \leq k$, since $\phi_j \left(x_{[l]}\right) = u_k \left(x_{[l]}\right) = y^+_{[l]}$ for such l. And of course (A.3) holds for $l = j$. So suppose $q \leq l < j$. Then

$$\phi_j \left(x_{[l]}\right) = y^-_{[j]} - \left(x_{[j]} - x_{[l]}\right) z^-_{[j]} \leq y^-_{[j]} - \left(x_{[j]} - x_{[l]}\right) \frac{y^-_{[j]} - y^+_{[l]}}{x_{[j]} - x_{[l]}} = y^+_{[l]}.$$

This is one of the two inequalities in (A.3). Now, by j-lower consistency we know there is some function $h_j \in \mathcal{H}_k$ having $h_j \left(x_{[j]}\right) = y^-_{[j]}$. If $z^-_{[j]} = 0$, then

$$y^-_{[l]} \leq h_j \left(x_{[l]}\right) \leq h_j \left(x_{[j]}\right) = y^-_{[j]} = \phi_j \left(x_{[l]}\right).$$

On the other hand, if $z^-_{[j]} > 0$, then

$$\begin{aligned}
y^-_{[l]} &\leq h_j \left(x_{[l]}\right) \\
&\leq \frac{x_{[j]} - x_{[l]}}{x_{[j]} - x_{[q]}} h_j \left(x_{[q]}\right) + \frac{x_{[l]} - x_{[q]}}{x_{[j]} - x_{[q]}} h_j \left(x_{[j]}\right) \\
&\leq \frac{x_{[j]} - x_{[l]}}{x_{[j]} - x_{[q]}} y^+_{[q]} + \frac{x_{[l]} - x_{[q]}}{x_{[j]} - x_{[q]}} h_j \left(x_{[j]}\right) \\
&= \frac{x_{[j]} - x_{[l]}}{x_{[j]} - x_{[q]}} y^+_{[q]} + \frac{x_{[l]} - x_{[q]}}{x_{[j]} - x_{[q]}} y^-_{[j]} \\
&= y^-_{[j]} - \frac{y^-_{[j]} - y^+_{[q]}}{x_{[j]} - x_{[q]}} \left(x_{[j]} - x_{[l]}\right) \\
&= \phi_j \left(x_{[l]}\right),
\end{aligned}$$

proving that (A.3) holds. We omit the similar argument establishing (A.3) for $j < l \leq r$, and conclude $\phi_j \in \mathcal{H}_k$.

Finally, we simply observe that $z^-_{[j]}, z^+_k[j] \in \partial\phi_j \left(x_{[j]}\right)$ follows from the definition of ϕ_j and the convexity of ϕ_j.

Proof of Proposition 4.6

Clearly the equality $\mathcal{H}_k(x) = [l_k(x), u_k(x)]$ implies consistency. So suppose that $(\mathbf{x}_k, \mathbf{y}^-_k, \mathbf{y}^+_k)$ is consistent. Let $h \in \mathcal{H}_k$ and $x \in [x_{[j]}, x_{[j+1]}]$ for some $j = 1, \ldots, k-1$. By convexity of h,

$$\begin{aligned} h(x) &\leq \frac{x - x_{[j]}}{x_{[j+1]} - x_{[j]}} h(x_{[j+1]}) + \frac{x_{[j+1]} - x}{x_{[j+1]} - x_{[j]}} h(x_{[j]}) \\ &\leq \frac{x - x_{[j]}}{x_{[j+1]} - x_{[j]}} y^+_{[j+1]} + \frac{x_{[j+1]} - x}{x_{[j+1]} - x_{[j]}} y^+_{[j]} \\ &= u_k(x). \end{aligned}$$

Now, if $z \in \partial h(x_{[j]})$, then Proposition 4.5 implies

$$h(x) \geq h(x_{[j]}) + (x - x_{[j]})z \geq y^-_{[j]} + (x - x_{[j]})z^-_{[j]}.$$

Similarly,

$$h(x) \geq y^-_{[j+1]} - (x_{[j+1]} - x)z^+_{[j]},$$

proving $h(x) \geq l_k(x)$.

Now we show that the bounds are tight, i.e., $l_k(x), u_k(x) \in \mathcal{H}_k(x)$. This, along with convexity of $\mathcal{H}_k$, suffices to prove the result.

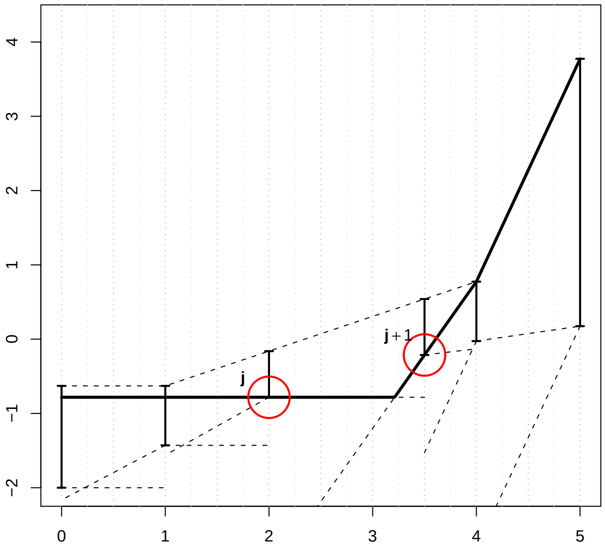
By Lemma 1, $u_k(x) \in \mathcal{H}_k(x)$. To finish the proof, we will exhibit a function in

$\mathcal{H}_k$ that agrees with l_k on the interval $[x_{[j]}, x_{[j+1]}]$. Let

$$\phi(x) = \begin{cases} \phi_j(x) & \text{if } x < x_{[j]}, \\ l_k(x) & \text{if } x_{[j]} \le x \le x_{[j+1]}, \\ \phi_{j+1}(x) & \text{if } x > x_{[j+1]}, \end{cases}$$

where ϕ_j, ϕ_{j+1} are as in Proposition 4.5; see Figure A.2.

Figure A.2: The function ϕ.



Clearly ϕ is increasing, piecewise linear, continuous, and satisfies $y^-_{[l]} \le \phi(x_{[l]}) \le y^+_{[l]}$ for all $l = 0, \ldots, k$. It is also clear that ϕ is convex on the intervals $[0, x_{[j]}]$, $[x_{[j]}, x_{[j+1]}]$, and $[x_{[j+1]}, x_{[k]}]$. Therefore, we need only check that the slope increases across the boundaries of the regions on which ϕ is defined.

But this fact follows immediately from the definitions: if $j > 0$, then

$$\frac{\phi(x_{[j]}) - \phi(x_{[j-1]})}{x_{[j]} - x_{[j-1]}} = \frac{\phi_j(x_{[j]}) - \phi_j(x_{[j-1]})}{x_{[j]} - x_{[j-1]}} = z^-_k[j],$$

which is less than or equal to the slope of l_k anywhere on $[x_{[j]}, x_{[j+1]}]$. Likewise, if

$j < k$, then

$$\frac{\phi(x_{[j+1]}) - \phi(x_{[j]})}{x_{[j+1]} - x_{[j]}} = \frac{\phi_{j+1}(x_{[j+1]}) - \phi_{j+1}(x_{[j]})}{x_{[j+1]} - x_{[j]}} = z_k^+[j+1],$$

which is greater than or equal to the slope of l_k anywhere on that same interval. Therefore, we conclude that ϕ is convex, and hence $\phi \in \mathcal{H}_k$.

Proof of Proposition 4.7

We prove

$$\mathcal{H}_{k+1} = \mathcal{H}_k \cap \{h : Y^- \leq h(x_{k+1}) \leq Y^+\}. \tag{A.4}$$

in steps 1–3; we prove that $\mathcal{H}_{k+1}$ is consistent (or empty) in steps 4–7.

Step One. Let us assume (inductively) that $(\mathbf{x}_k, \mathbf{y}_k^-, \mathbf{y}_k^+)$ is consistent. It is immediately seen from Algorithms 3.1 and 3.2 of Chapter 2 that $l_k(x_{[j]}) \leq y_{[j]}^-$ and $y_{[j]}^+ \leq u_k(x_{[j]})$ for all $j = 1, \ldots, k+1$. Therefore, $\mathcal{H}_{k+1} \subset \mathcal{H}_k$. It is also immediate that $Y^- \leq y_{k+1}^-$ and $y_{k+1}^+ \leq Y_k^+$, and so we have

$$\mathcal{H}_{k+1} \subset \mathcal{H}_k \cap \{h : Y^- \leq h(x_{k+1}) \leq Y^+\}.$$

For the rest of the proof, we assume that $\mathcal{H}_k \cap \{h : Y^- \leq h(x_{k+1}) \leq Y^+\}$ is nonempty (for otherwise the remaining claims are true vacuuously).

Step Two. Suppose $h \in \mathcal{H}_k$ and $Y^- \leq h(x_{k+1}) \leq Y^+$. In this step, we prove that $h(x_{[j]}) \leq y_{[j]}^+$ for $j = 1, \ldots, k+1$. For the remainder of the proof, let i be the order statistic of the point inserted at the $(k+1)^{\text{th}}$ iteration, so that $(x_{k+1}, y_{k+1}^-, y_{k+1}^+) = (x_{[i]}, y_{[i]}^-, y_{[i]}^+)$.

Consider the main conditional expression in Algorithm 3.1 of Chapter 2. If $Y^+ \geq u_k(x_{k+1})$, then we have

$$y_{[j]}^+ = u_k(x_{[j]}) \geq h(x_{[j]})$$

for all $j = 1, \dots, k+1$, and upper consistency of $(\mathbf{x}_{k+1}, \mathbf{y}_{k+1}^-, \mathbf{y}_{k+1}^+)$ follows from upper consistency of $(\mathbf{x}_k, \mathbf{y}_k^-, \mathbf{y}_k^+)$.

So suppose instead that $Y^+ < u_k(x_{k+1})$. Then

$$y_{[i]}^+ = Y^+ \geq h(x_{[i]}) \tag{A.5}$$

by assumption. We now proceed to show by (backward) induction that

$$h(x_{[j]}) \leq y_{[j]}^+ \tag{A.6}$$

for all $j = k+1, \dots, i+1$. For the base case ($j = k+1$), we have

$$\begin{aligned} h(x_{[k+1]}) &\leq h(x_{[i]}) + (x_{[k+1]} - x_{[i]})c \quad (\text{since } h \in \mathcal{H}_0) \\ &\leq Y^+ + (x_{[k+1]} - x_{[i]})c. \end{aligned}$$

Furthermore, $h(x_{[k+1]}) \leq u_k(x_{[k+1]}) = \tilde{y}_{[k]}^+$. Therefore, by line 5 of Algorithm 3.1 of Chapter 2, we conclude that (A.6) holds in the base case. Now, for $j = k, \dots, i+1$, we have $h(x_{[j]}) \leq u_k(x_{[j]}) = \tilde{y}_{[j-1]}^+$ and

$$\begin{aligned} h(x_{[j]}) &\leq \frac{x_{[j+1]} - x_{[j]}}{x_{[j+1]} - x_{[i]}} h(x_{[i]}) + \frac{x_{[j]} - x_{[i]}}{x_{[j+1]} - x_{[i]}} h(x_{[j+1]}) && \text{(by convexity)} \\ &\leq \frac{x_{[j+1]} - x_{[j]}}{x_{[j+1]} - x_{[i]}} Y^+ + \frac{x_{[j]} - x_{[i]}}{x_{[j+1]} - x_{[i]}} y_{[j+1]}^+ && \text{(by (A.5) and induction)} \\ &= Y^+ + \frac{y_{[j+1]}^+ - Y^+}{x_{[j+1]} - x_{[i]}} (x_{[j]} - x_{[i]}). \end{aligned}$$

Therefore, line 7 of Algorithm 3.1 of Chapter 2 implies (A.6) holds for $j = i, \dots, k+1$. We omit the similar proof that (A.6) holds for $j = 1, \dots, i-1$.

Step Three. In this step, we show by induction that

$$h(x_{[j]}) \geq y_{[j]}^-. \tag{A.7}$$

for $j = i, \dots, k+1$, omitting the similar proof that (A.7) holds for $j = 1, \dots, i-1$.

We have $h(x_{[i]}) \geq Y^-$ by assumption and $h(x_{[i]}) \geq l_k(x_{[i]})$ by Proposition 4.5. So by Algorithm 3.2 of Chapter 2, line 1, (A.7) holds for $j = i$.

Now suppose (inductively) that $h(x_{[j-1]}) \geq y^-_{[j-1]}$ for $j - 1 \geq i$. By line 4 of Algorithm 3.2 of Chapter 2, we know that either $y^-_{[j]} = \tilde{y}^-_{[j-1]}$ or

$$y^-_{[j]} = y^-_{[j-1]} + (x_{[j]} - x_{[j-1]})z^-_{[j-1]}. \tag{A.8}$$

In the former case, the fact that $h \in \mathcal{H}_k$ implies the desired inequality. Therefore, let us assume (A.8) holds. Suppose $z^-_{[j-1]} = 0$. Then the fact that h is nondecreasing and the inductive hypothesis imply (A.7). So suppose instead that $z^-_{[j-1]} > 0$. In this case, by construction there must be some $l < j - 1$ such that

$$z^-_{[j-1]} = \frac{y^-_{[j-1]} - y^+_{[l]}}{x_{[j-1]} - x_{[l]}} \leq \frac{h(x_{[j-1]}) - h(x_{[l]})}{x_{[j-1]} - x_{[l]}}. \tag{A.9}$$

Here, the inequality follows from the induction hypothesis and from Step Two of this proof. Combining (A.8) and (A.9) and invoking the convexity of h yields

$$\begin{aligned} y^-_{[j]} &\leq y^-_{[j-1]} + \frac{h(x_{[j-1]}) - h(x_{[l]})}{x_{[j-1]} - x_{[l]}}(x_{[j]} - x_{[j-1]}) \\ &\leq y^-_{[j-1]} + \frac{h(x_{[j]}) - h(x_{[j-1]})}{x_{[j]} - x_{[j-1]}}(x_{[j]} - x_{[j-1]}) \\ &= h(x_{[j]}). \end{aligned}$$

Therefore, (A.7) holds. We conclude that $h \in \mathcal{H}_{k+1}$, proving (A.4).

Step Four. We prove that the new state $(\mathbf{x}_{k+1}, \mathbf{y}^-_{k+1}, \mathbf{y}^+_{k+1})$ is upper consistent.

If $Y^+ \geq u_k(x_{k+1})$, then upper consistency of $(\mathbf{x}_{k+1}, \mathbf{y}^-_{k+1}, \mathbf{y}^+_{k+1})$ follows from (A.4) and upper consistency of $(\mathbf{x}_k, \mathbf{y}^-_k, \mathbf{y}^+_k)$, since in this case we have $u_k = u_{k+1} \in \mathcal{H}_{k+1}$. Suppose instead that $Y^+ < u_k(x_{k+1})$. We will show that $u_{k+1} \in \mathcal{H}_{k+1}$ in this case as well. By construction, $y^+_{[j]} = u_{k+1}(x_{[j]}) \geq y^-_{[j]}$ for all $j = 1, \ldots, k+1$. So we need only prove $u_{k+1} \in \mathcal{H}_0$, i.e., that u_{k+1} is increasing and convex and satisfies the bounded growth condition. Since u_{k+1} is piecewise linear, we need only check that these conditions hold at its knots, i.e.,

$$\begin{aligned} 0 \leq \frac{y^+_{[j+1]} - y^+_{[j]}}{x_{[j+1]} - x_{[j]}} &\leq c, \quad \text{for } j = 1, \ldots, k+1, \text{ and} \\ \frac{y^+_{[j]} - y^+_{[j-1]}}{x_{[j]} - x_{[j-1]}} \leq \frac{y^+_{[j+1]} - y^+_{[j]}}{x_{[j+1]} - x_{[j]}}&, \qquad\qquad \text{for } j = 1, \ldots, k. \end{aligned} \tag{A.10}$$

We provide details of the proof of these statements for $j \in \{i, \ldots, k+1\}$; the proof for $j \in \{1, \ldots, i-1\}$ is similar and is thus omitted. From line 5 of Algorithm 3.1 of Chapter 2, either

$$y^+_{[k+1]} = \tilde{y}^+_{[k]} = u_k(x_{[k+1]}) \geq u_k(x_{k+1}) \geq Y^+ = y^+_{[i]}$$

or

$$y^+_{[k+1]} = y^+_{[i]} + (x_{[k+1]} - x_{[i]})c \geq y^+_{[i]}.$$

It follows from line 7 of the algorithm and an easy induction argument that

$$y^+_{[i]} \leq y^+_{[j]}$$

for all $j = i, \ldots, k+1$. Now (A.10) follows directly from lines 5 and 7.

Step Five. We prove lower consistency. It follows from (A.4) and Proposition 4.6 (applied to $(\mathbf{x}_k, \mathbf{y}^-_k, \mathbf{y}^+_k)$, which we have assumed to be consistent) that $(\mathbf{x}_{k+1}, \mathbf{y}^-_{k+1}, \mathbf{y}^+_{k+1})$ is i-lower consistent. Now, let $j \geq i$ and assume (inductively on j) that $(\mathbf{x}_{k+1}, \mathbf{y}^-_{k+1}, \mathbf{y}^+_{k+1})$ is j-lower consistent. (We will omit the similar induction proof that $(\mathbf{x}_{k+1}, \mathbf{y}^-_{k+1}, \mathbf{y}^+_{k+1})$ is lower consistent at indices less than i.) Let r be the least index in $\{j+1, \ldots, k+1\}$ such that $y^-_{[r]} = \tilde{y}^-_{[r-1]}$ (as in line 5 of Algorithm 3.1 of Chapter 2), or let $r = k+2$ if there is no such index. We claim that this definition of r implies

$$\begin{aligned} y^-_{[l-1]} &= y^-_{[j]} + (x_{[l-1]} - x_{[j]})z^-_{[j]}, \text{ and} \\ z^-_{[l-1]} &= z^-_{[j]} \end{aligned} \tag{A.11}$$

for all $l = j+1, \ldots, r$. Clearly, (A.11) holds for $l = j+1$. Assume inductively that (A.11) holds for a particular $l \in j+1, \ldots, r$. Then by the definition of r and by lines 5 and 6 of Algorithm 3.1 of Chapter 2,

$$\begin{aligned} y^-_{[l]} &= y^-_{[l-1]} + (x_{[l]} - x_{[l-1]})z^-_{[l-1]} \\ &= y^-_{[j]} + (x_{[l]} - x_{[j]})z^-_{[j]} \end{aligned}$$

and

$$z^-_{[j]} \leq z^-_{[l]} = z^-_{[j]} \vee \frac{y^-_{[l]} - y^+_{[l-1]}}{x_{[l]} - x_{[l-1]}} \leq z^-_{[j]} \vee \frac{y^-_{[l]} - y^-_{[l-1]}}{x_{[l]} - x_{[l-1]}} = z^-_{[j]}.$$

Therefore, (A.11) holds for all $l = j+1, \ldots, r$.

Let $\phi^{k+1}_j \in \mathcal{H}_{k+1}$ and $\phi^k_{r-1} \in \mathcal{H}_k$ be as in Proposition 4.5. We are able to invoke that proposition for ϕ^{k+1}_j using the inductive hypothesis that $(\mathbf{x}_{k+1}, \mathbf{y}^-_{k+1}, \mathbf{y}^+_{k+1})$ is j-lower consistent, and we are able to invoke it for ϕ^k_{r-1} using the inductive hypothesis that $(\mathbf{x}_k, \mathbf{y}^-_k, \mathbf{y}^+_k)$ is consistent.

Now define

$$\psi_j(x) = \begin{cases} \phi^{k+1}_j(x) & \text{if } x_{[1]} \leq x < x_{[j]}, \\ y^-_{[j]} + (x - x_{[j]})z^-_{[j]} & \text{if } x_{[j]} \leq x \leq x_{[r-1]}, \\ y^-_{[r]} - \frac{y^-_{[r]} - y^-_{[r-1]}}{x[r] - x_{[r-1]}}(x_{[r]} - x) & \text{if } r \leq k+1 \text{ and } x_{[r-1]} < x \leq x_{[r]}, \\ \phi^k_{r-1}(x) & \text{if } r \leq k \text{ and } x_{[r]} < x \leq x_{[k+1]}. \end{cases}$$

We will show that $\psi_j \in \mathcal{H}_{k+1}$ and that $\psi_j(x_{[j+1]}) = y^-_{[j+1]}$, thus demonstrating $(j+1)$-lower consistency of $\mathcal{H}_{k+1}$. The fact that $\phi^{k+1}_j \in \mathcal{H}_{k+1}$ implies that $Y^- \leq$

$\psi_j(x_{[i]}) = \phi_j^{k+1}(x_{[i]}) \leq Y^+$. Therefore, by (A.4), we need only show $\psi_j \in \mathcal{H}_k$ in order to prove $\psi_j \in \mathcal{H}_{k+1}$.

Step Six. In this step we show that

$$\psi_j \text{ is continuous,} \tag{A.12}$$

$$\tilde{y}^-_{[l]} \leq \psi_j(\tilde{x}_{[l]}) \leq \tilde{y}^+_{[l]} \text{ for all } l \in \{1, \ldots, k\}, \tag{A.13}$$

$$\psi_j(x_{[j+1]}) = y^-_{[j+1]}. \tag{A.14}$$

Observe that (A.12) is evident at all points x except possibly $x = x_{[r-1]}$, so we may replace (A.12) by

$$\psi_j \text{ is continuous at } x = x_{[r-1]}. \tag{A.15}$$

Also, (A.13) holds automatically for $l = 1, \ldots, j-1$ and $l = r, \ldots, k$, so it need only be checked for $l = j, \ldots, r-1$.

The facts (A.13), (A.14), and (A.15) are automatic from the definition of ψ_j if $r = j+1$, so assume $r > j+1$. Then by (A.11),

$$y^-_{[r-1]} = y^-_{[j]} + (x_{[r-1]} - x_{[j]})z^-_{[j]},$$

proving (A.15),

$$\psi_j(x_{[j+1]}) = y^-_{[j]} + (x_{[j+1]} - x_{[j]})z^-_{[j]} = y^-_{[j+1]},$$

proving (A.14), and for $l = j, \ldots, r-1$,

$$\psi_j(\tilde{x}_{[l]}) = \psi_j(x_{[l+1]}) = y^-_{[j]} + (x_{[l+1]} - x_{[j]})z^-_{[j]} = y^-_{[l+1]} \in [\tilde{y}^-_{[l]}, \tilde{y}^+_{[l]}],$$

proving (A.13).

Step Seven. All that remains is to prove convexity of ψ_j, as the monotonicity and bounded growth conditions are clear from the construction of ψ_j. Evidently,

ψ_j is convex on the interval $[x_{[1]}, x_{[r-1]}]$, which follows from the construction of ϕ_j^{k+1}. Also, if $r \leq k$, then ψ_j is (separately) convex on $[x_{[r]}, x_{[k+1]}]$. Therefore, in order to prove ψ_j is convex, we need prove only two inequalities:

$$z_{[j]}^{-} \leq \frac{y_{[r]}^{-} - y_{[r-1]}^{-}}{x_{[r]} - x_{[r-1]}}, \qquad \text{if } r \leq k+1, \text{ and} \tag{A.16}$$

$$\frac{y_{[r]}^{-} - y_{[r-1]}^{-}}{x_{[r]} - x_{[r-1]}} \leq \tilde{z}_{[r-1]}^{+}, \qquad \text{if } r \leq k. \tag{A.17}$$

First, for $r \leq k+1$, we have

$$\frac{y_{[r]}^{-} - y_{[r-1]}^{-}}{x_{[r]} - x_{[r-1]}} \geq z_{[r-1]}^{-} = z_{[j]}^{-},$$

proving (A.16). Second, for $r \leq k$, by the definition of r,

$$\begin{aligned} \frac{y_{[r]}^{-} - y_{[r-1]}^{-}}{x_{[r]} - x_{[r-1]}} &= \frac{l_k(\tilde{x}_{[r-1]}) - y_{[r-1]}^{-}}{\tilde{x}_{[r-1]} - x_{[r-1]}} \\ &\leq \frac{l_k(\tilde{x}_{[r-1]}) - l_k(x_{[r-1]})}{\tilde{x}_{[r-1]} - x_{[r-1]}} \\ &\leq \tilde{z}_{[r-1]}^{+}, \end{aligned}$$

as desired.

BIBLIOGRAPHY

Andersen, L. and M. Broadie (2004). Primal-dual simulation algorithm for pricing multidimensional American options. *Management Sci. 50*(9), 1222–1234.

Asmussen, S. and P. W. Glynn (2007). *Stochastic simulation: algorithms and analysis*, Volume 57 of *Stochastic modelling and applied probability*. Springer.

Avramidis, A. N. and J. R. Wilson (1993). Integrated variance reduction strategies. In *WSC '93: Proceedings of the 25th conference on Winter simulation*, New York, NY, USA, pp. 445–454. Institute of Electrical and Electronics Engineers, Inc.: ACM.

Bastin, F., C. Cirillo, and P. L. Toint (2006). Convergence theory for nonconvex stochastic programming with an application to mixed logit. *Mathematical Programming B 108*, 207–234.

Bolia, N. and S. Juneja (2005). Function-approximation-based perfect control variates for pricing American options. In M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Jones (Eds.), *Proceedings of the Winter Simulation Conference*, Piscataway, New Jersey. Institute of Electrical and Electronics Engineers, Inc.

Broadie, M. and P. Glasserman (2004). A stochastic mesh method for pricing high-dimensional American options. *Journal of Computational Finance 7*(4), 35–72.

Carriere, J. F. (1996). Valuation of the early-exercise price for options using simulations and nonparametric regression. *Insurance: Mathematics and Economics 19*(1), 19–30.

Chen, H. and B. Schmeiser (2001). Stochastic root finding via retrospective approximation. *IIE Transactions 33*(3), 259–275.

Chen, V. C. P., D. Ruppert, and C. A. Shoemaker (1999). Applying experimental design and regression splines to high-dimensional continuous-state stochastic dynamic programming. *Oper. Res. 47*(1), 38–53.

Clément, E., D. Lamberton, and P. Protter (2002). An analysis of a least squares regression method for American option pricing. *Finance Stoch. 6*(4), 449–471.

Cox, J. C., S. A. Ross, and M. Rubinstein (1979). Option pricing: A simplified approach. *Journal of Financial Economics 7*, 229–263.

den Boef, E. and D. den Hertog (2007). Efficient line search methods for convex functions. *SIAM Journal on Optimization 18*(1), 338–363.

Devroye, L. (1990). Coupled samples in simulation. *Operations Research 38*, 115–126.

Duffie, D. (2001). *Dynamic Asset Pricing Theory* (Third ed.). Princeton, New Jersey: Princeton University Press.

Edelman, A., T. A. Arias, and S. T. Smith (1999). The geometry of algorithms with orthogonality constraints. *SIAM journal on matrix analysis and applications 20*(2), 303–353.

Ehrlichman, S. M. T. and S. G. Henderson (2007a). Adaptive control variates for pricing multi-dimensional American options. *Journal of Computational Finance 11*(1), 65–91.

Ehrlichman, S. M. T. and S. G. Henderson (2007b). Finite-sample performance guarantees for one-dimensional stochastic root finding. In S. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton (Eds.), *Proceedings of the 2007 Winter Simulation Conference*, Piscataway, New Jersey. Institute of Electrical and Electronics Engineers, Inc.

Ehrlichman, S. M. T. and S. G. Henderson (2008). Comparing two systems: Beyond common random numbers. In S. Mason, R. Hill, L. Moench, and

O. Rose (Eds.), *Proceedings of the 2008 Winter Simulation Conference*, Piscataway, New Jersey. Institute of Electrical and Electronics Engineers, Inc.

Ekström, E. (2004). Properties of American option prices. *Stochastic Processes and Their Applications 114*(2), 265–278.

Fisher, R. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics 7*(2), 179–188.

Fishman, G. (1996). *Monte Carlo: Concepts, Algorithms, and Applications.* Springer New York.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *Ann. Statist. 19*(1), 1–141. With discussion and a rejoinder by the author.

Friedman, J. H., E. Grosse, and W. Stuetzle (1983). Multidimensional additive spline approximation. *SIAM J. Sci. Statist. Comput. 4*(2), 291–301.

Friedman, J. H. and W. Stuetzle (1981). Projection pursuit regression. *J. Amer. Statist. Assoc. 76*(376), 817–823.

Glasserman, P. (2004). *Monte Carlo Methods in Financial Engineering*, Volume 53 of *Applications of Mathematics.* New York: Springer-Verlag.

Glasserman, P. and D. D. Yao (1992). Some guidelines and guarantees for common random numbers. *Management Science 38*, 884–908.

Glasserman, P. and D. D. Yao (2004). Optimal couplings are totally positive and more. *Journal of Applied Probability 41*, 321–332.

Glasserman, P. and B. Yu (2004). Simulation for American options: regression now or regression later? In *Monte Carlo and Quasi-Monte Carlo Methods 2002*, pp. 213–226. Berlin: Springer.

Grant, D., G. Vora, and D. Weeks (1996). Simulation and the Early-Exercise Option Problem. *Journal of Financial Engineering 5*(3), 211–227.

Gross, O. and S. Johnson (1959). Sequential Minimax Search for a Zero of a Convex Function. *Mathematical Tables and Other Aids to Computation 13*(65), 44–51.

Hammersley, J. M. and D. C. Handscomb (1964). *Monte Carlo Methods*. New York: Wiley.

Hastie, T., R. Tibshirani, and J. H. Friedman (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer Verlag.

Haugh, M. B. and L. Kogan (2004). Pricing American options: a duality approach. *Oper. Res. 52*(2), 258–270.

Henderson, S. G. (2006). Mathematics for simulation. In S. G. Henderson and B. L. Nelson (Eds.), *Simulation*, Volume 13 of *Handbooks in Operations Research and Management Science*. Amsterdam: Elsevier.

Henderson, S. G. and P. W. Glynn (2002). Approximating martingales for variance reduction in Markov process simulation. *Math. Oper. Res. 27*(2), 253–271.

Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies 6*(2), 327–43. available at http://ideas.repec.org/a/oup/rfinst/v6y1993i2p327-43.html.

Horn, R. A. and C. R. Johnson (1985). *Matrix Analysis*. Cambridge, UK: Cambridge University Press.

Jones, L. K. (1987). On a conjecture of Huber concerning the convergence of projection pursuit regression. *Ann. Statist. 15*(2), 880–882.

Kelton, W. D. (2006). Implementing representations of uncertainty. In S. G. Henderson and B. L. Nelson (Eds.), *Simulation*, Volume 13 of *Handbooks in Operations Research and Management Science*. Amsterdam: Elsevier.

Kim, S.-H. and B. L. Nelson (2006). Selecting the best system. In S. G. Henderson and B. L. Nelson (Eds.), *Simulation*, Volume 13 of *Handbooks in*

Operations Research and Management Science. Amsterdam: Elsevier.

Kočvara, M. and M. Stingl (2003, June). PENNON: a code for convex nonlinear and semidefinite programming. *Optimization Methods and Software 18*(3), 317–333.

Kushner, H. and G. Yin (2003). *Stochastic approximation algorithms and applications* (second ed.). Springer.

Laprise, S. B., M. C. Fu, S. I. Marcus, A. E. B. Lim, and H. Zhang (2006, January). Pricing American-style derivatives with European call options. *Management Scuence 52*(1), 95–110.

Leisch, F., K. Hornik, and B. D. Ripley (2005). *mda: Mixture and flexible discriminant analysis.* R Foundation for Statistical Computing. R package version 0.3-1.

Lindvall, T. (1992). *Lectures on the Coupling Method*. New York: Wiley.

Lippert, R. A. and A. Edelman (1999). sgmin [computer software]. Available at http://www-math.mit.edu/ lippert/sgmin.html.

Longstaff, F. A. and E. S. Schwartz (2001). Valuing American options by simulation: A simple least-squares approach. *Review of Financial Studies 14*(1), 113–47.

Marsaglia, G. and I. Olkin (1984). Generating Correlation Matrices. *SIAM Journal on Scientific and Statistical Computing 5*, 470.

Pasupathy, R. and B. W. Schmeiser (2003). Some issues in multivariate stochastic root finding. In S. Chick, P. Sanchez, D. Ferrin, and D. Morrice (Eds.), *Proceedings of the 2003 Winter Simulation Conference*, Piscataway, New Jersey. Institute of Electrical and Electronics Engineers, Inc.

Pasupathy, R. and B. W. Schmeiser (2004). Retrospective approximation algorithms for the multidimensional stochastic root-finding problem. In R. Ingalls, M. Rossetti, J. Smith, and B. Peters (Eds.), *Proceedings of the 2004*

Winter Simulation Conference, Piscataway, New Jersey. Institute of Electrical and Electronics Engineers, Inc.

Potra, F. (1994). Efficient hybrid algorithms for finding zeros of convex functions. *Journal of Complexity 10*(2), 199–215.

Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling (1992). *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press.

R Development Core Team (2005). *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing. ISBN 3-900051-07-0.

Rasmussen, N. S. (2005). Control variates for Monte Carlo valuation of American options. *J. Comp. Finance 9*(1), 84–102.

Rogers, L. C. G. (2002). Monte Carlo valuation of American options. *Math. Finance 12*(3), 271–286.

Rote, G. (1992). The convergence rate of the sandwich algorithm for approximating convex functions. *Computing 48*(3), 337–361.

Rudin, W. (1987). *Real and Complex Analysis* (Third ed.). McGraw-Hill series in higher mathematics. Boston, MA: McGraw-Hill.

Schmeiser, B. and V. Kachitvichyanukul (1986). Correlation induction without the inverse transformation. In J. Wilson, J. Henriksen, and S. Roberts (Eds.), *Proceedings of the 1986 Winter Simulation Conference*, New York, NY, pp. 266–274. ACM.

Shapiro, A. (2004). Monte Carlo sampling methods. In A. Ruszczynski and A. Shapiro (Eds.), *Stochastic Programming*, Volume 10 of *Handbooks in Operations Research and Management Science.* Amsterdam: Elsevier.

Spall, J. (2003). *Introduction to Stochastic Search and Optimization.* John Wiley & Sons, Inc. New York, NY, USA.

Thorisson, H. (2000). *Coupling, Stationarity, and Regeneration*. New York, NY: Springer.

Tsitsiklis, J. N. and B. Van Roy (2001, July). Regression methods for pricing complex American-style options. *IEEE-NN 12*, 694–703.

Wald, A. (1947). *Sequential Analysis*. Chapman & Hall, Ltd.

Wright, R. D. and J. Ramsay, T. E. (1979). On the effectiveness of common random numbers. *Management Science 25*(7), 649–656.

Zhang, H., C.-Y. Yu, H. Zhu, and J. Shi (2003). Identification of linear directions in multivariate adaptive spline models. *J. Amer. Statist. Assoc. 98*(462), 369–376.