Journal of Statistical Planning and Inference 33 (1992) 295-309
North-Holland

Machine learning for optimal blackjack counting strategies

S. Yakowitz and M. Kollier

Systems and Industrial Engineering Department, University of Arizona, Tucson, AZ, USA

Received 28 September 1990; revised manuscript received 23 September 1991
Recommended by S. Zacks

Abstract: Search for optimal counting strategies for blackjack, like many other tasks in artificial intelligence, can be transcribed into the task of finding a global minimum of a multi-modal discontinuous objective function on the basis of noisy measurements. Herein we relate an algorithm for such 'stochastic minimization' problems, and derive some general properties. The general framework is extended to be applicable to finding the 'ten-count' standing-number strategy for blackjack. Computational experiments explore this learning application. The feature here, as opposed to blackjack analyses by Thorp, Braum, and others, who rely on combinatorics in their constructions, is that our methodology requires minimal modelling or understanding of the problem structure. This is the first reasonably detailed account of a procedure for finding ten-count standing numbers. But it is general enough to be applicable to other blackjack problems and, in fact, to a wide variety of sequential decision tasks.

AMS Subject Classifications: 62C25, 62G99, 62L12.

Key words: Automatic learning; games; sequential decisions.

1. Background and intentions

Automatic learning is a tantalizing goal extolled in an earnest and eloquent way by the early greats of computer science (e.g., Bellman (1978), von Neumann (1958), Wiener (1961)). These writers spoke of machine learning as the avenue by which the computer will come into its fullest powers, through devising and testing strategies unforeseen by its programmers. A statement of this genre, by Feigenbaum and Feldman (1963, p. 4), is as follows:

> ... it is wrong to conclude that a computer can exhibit behavior no more intelligent than its human programmer and that this astute gentleman can accurately predict the behavior of his program. These conclusions ignore the enormous complexity of information processing possible in problem-solving and learning machines.

Correspondence to: Sid Yakowitz, Systems and Industrial Engineering Department, University of Arizona, Tucson, AZ 85721, USA.

* Supported in part by NSF grant ECS-8913642.

The 1960's saw some specific machine learning case studies such as a reasonably successful checkers program (Samuel (1963)), and some investigations in simultaneous identification and control of linear systems (see, for instance, the review by Fu (1970)).

Under the banners of neural nets, simulated annealing, and genetic algorithms, there is now a flourish of activity in automatic learning. The first author (among others) has followed a divergent methodological path, the origins of which can be traced at least as far back as the nonparametric sequential design studies of Robbins and his colleagues (especially, Robbins (1952)). More will be said about this research direction in Section 2.

Being wary of applications to big, unintuitive problems, we have proceeded modestly with algorithm testing and case studies. Such experimentation includes the eight-puzzle (Yakowitz and Lugosi (1990)), go-moku (Yakowitz (1989)), queuing network tuning (Yakowitz et al. (1992a)) and epidemic control (Yakowitz et al. (1992b)). The learning program is assigned to solve a sequence of games or puzzles which are statistically similar in some sense. The effect of learning is exhibited, as more and more games are played, in that the computer achieves improving levels of performance and wins with ever-increasing regularity.

In this spirit, the present paper relates our exploration into the famous casino game, blackjack. This exploration is much inspired by Thorp's (1966) book, *Beat the Dealer*. Whereas Baldwin et al. (1956) proposed a statistical approach which supplied groundwork, Thorp's study is credited with providing the first published strategy to give a positive expected return to the player.

The point of our experiment into machine learning for blackjack is to see whether Thorp's 'ten-count' standing number table can be approximated by a general-purpose learning algorithm in conjunction with a blackjack simulator. This experiment will be considered successful if the approximation is good enough to yield about the same average return as that achieved by Thorp's standing numbers, under simulations of a one-deck blackjack game. As mentioned, the larger aim of this experiment is to put in place another stepping stone on the path to convincing ourselves and others that automatic learning has some practical merit, as a field of inquiry.

Section 2 is devoted to revealing our learning algorithm, which is shown to attain certain convergence properties which are as strong as any alternative known to us. Section 3 describes the application and extension of the general theory to the counting strategy learning problem. Section 4 sets forth the results of our computational blackjack learning experiment.

The central aims of this work, therefore, are (1) to describe and analyze a learning method, the convergence properties of which are, to our knowledge, the strongest to date, and (2) to demonstrate that this method is effective for finding a critical part of Thorp's strategy, which we hold to be among the most inspiring decision problem solutions in the literature. A point important to us is that we obtain the standing-number table without accounting for mathematical structure. We do not know how Thorp and his collaborators obtained their standing numbers; as referenced in Section 3, indications are that it was through intricate combinatorial schemes and approximations which were patently specific for blackjack.

2. Automatic learning

2.1. A statistical framework

As we will see, search for an optimal blackjack strategy (like a great many other learning problems in artificial intelligence) can be transcribed to a certain statistical task which we refer to as the 'stochastic minimization problem'. The components are

- a domain D (of 'decisions') in a Euclidean space,
- a bounded, measurable real-valued function $f(x)$, $x \in D$, to be minimized, and
- a random noise $W(x)$ with decision-dependent distribution.

Initially, only D is known to the decision maker. The *stochastic minimization problem* is the task of choosing, at each decision epoch n, a point $x(n)$ so that in some sense,

$$f(x(n)) \to f_{\min}, \tag{2.1}$$

$f_{\min}$ being the global minimum of $f(\cdot)$. The choice of $x(n)$ may be a function of earlier data $\{(x(i), y(i))\}_{i<n}$. The values $y(i)$ are noisy function observations,

$$y(i) = f(x(i)) + w(x(i)). \tag{2.2}$$

The noise will be presumed to depend on earlier observations only through the choice of $x(n)$. The stochastic minimization problem is related to problems of stochastic approximation, except here we make no assumptions about unimodality or smoothness of the objective function.

Under nomenclature such as 'random search', there are several techniques in the literature (e.g., Devroye (1978a,b), Gurin (1966), Matyas (1965), Yakowitz and Fisher (1973)) which are known to solve the stochastic minimization problem in one sense or another. Our recent efforts (Yakowitz and Lugosi (1990), Yakowitz and Lowe (1991)) have been in the direction of learning through simultaneous optimization and simulation. The method in the first study assures convergence in probability under a moment condition and the second draws connections with (infinite-armed) bandit problems and gives rates. These last two works ignored much sample information in order to sidestep analytic difficulties in showing properties of stopping times. A feature of the method below is that it uses all the sample information and thus is likely to be more accurate in the early stages of the learning process.

Even though the present task of recovering the standing number table for blackjack is 'off line', the use of our machine learning algorithm makes sense. By concentrating observations at more promising points, it assures that good decisions are examined carefully to make sure that the performance is close to the theoretical expectation and not simply the result of a lucky streak. Similarly, unpromising ones are neglected except for very occasional samples.

2.2. An algorithm for the stochastic minimization problem

2.2.1. Some algorithm components

(i) $p(x)$ is any probability density function having as support all of a decision space D.

(ii) $\{\mathrm{NP}(n)\}$ and $\{N(n)\}$ are nondecreasing unbounded sequences of integers such that $N(1)=\mathrm{NP}(1)=1$, $\mathrm{NP}(0)=0$, and

$$\mathrm{NP}(n)N(n)/n \downarrow 0. \tag{2.3}$$

Set $n=1$ and proceed to the iterative step.

2.2.2. The iterative step

New Sample Times (*Steps* 1 *and* 2). If $\mathrm{NP}(n-1)<\mathrm{NP}(n)$, select a new point from D.

1. Choose a point, designated by $t(\mathrm{NP}(n))$, from D at random according to the pdf $p(x)$. Set $x(n)=t(\mathrm{NP}(n))$ and observe $y(n)$.

2. Start a running average $m_{\mathrm{NP}(n)}$ for observations at $t(\mathrm{NP}(n))$ by declaring

$$m_{\mathrm{NP}(n)}=y(n),$$

and start a counter at $t(\mathrm{NP}(n))$ by setting $\mathrm{NS}(\mathrm{NP}(n))=1$. (Thus you see that '$\mathrm{NP}(n)$' tells the number of New [sample] Points that have been gathered by time n, and '$\mathrm{NS}(j)$' gives the Number of Samples that have been taken at a given sample point $t(j)$, $j\leqslant \mathrm{NP}(n)$.) Sometimes the argument n will be omitted.

Resample Times (*Steps* 3 *and* 4). Else if $\mathrm{NP}(n-1)=\mathrm{NP}(n)$, resample at the apparently best point.

3. Let I^* be any index i, $1\leqslant i\leqslant \mathrm{NP}(n)$, such that

$$m_{I^*}\leqslant m_j, \qquad 1\leqslant j\leqslant \mathrm{NP}(n). \tag{2.4}$$

Set $x(n)=t(I^*)$, and observe $y(n)$.

4. Update the sample mean.

Set

$$m_{I^*}=[m_{I^*}\cdot \mathrm{NS}(I^*)+y(n)]/(\mathrm{NS}(I^*)+1), \tag{2.5}$$

$$\mathrm{NS}(I^*)=\mathrm{NS}(I^*)+1. \tag{2.6}$$

Assure at least $N(n)$ observations at all points.

5. Skip this step if $\mathrm{NP}(n+1)>\mathrm{NP}(n)$. If, for some $j\leqslant \mathrm{NP}(n)$, $\mathrm{NS}(j)<N(n)$, set $n=n+1$, set $x(n)=t(j)$, observe $y(n)$, and update m_j according to

$$m_j=[m_j\cdot \mathrm{NS}(j)+y(n)]/(\mathrm{NS}(j)+1) \tag{2.7}$$

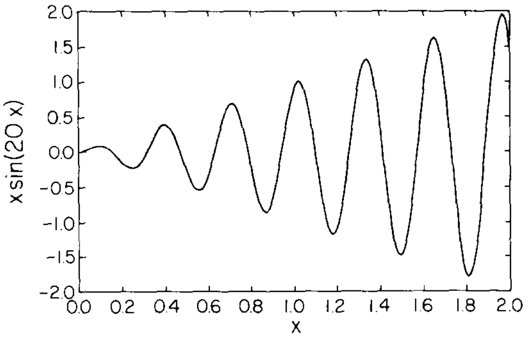

Fig. 1. A wiggly objective function.

and set $\mathrm{NS}(j) = \mathrm{NS}(j) + 1$. Repeat this step as necessary. Call these times *n retest* times.

6. Set $n = n + 1$. Repeat the iterative step. (There is no stopping condition.)

2.3. An experimental example

To illustrate the search, we applied it to the highly oscillatory function

$$f(x) = x \sin(20x)$$

on the domain $D = [0, 2]$. This function is plotted in Figure 1.

The noise $W(x)$ was Gaussian, with mean 0 and variance 1. Our algorith took $N(n) = \mathrm{NP}(n) = \mathit{IntegerPart}(n^{49/100})$. The random points $t(i)$ were chosen uniformly on D. Table 1 gives the average observed performance at successive readout times in the learning process. The minimal value of $f(x)$ on $[0, 2]$ is about -1.8.

2.4. A convergence theorem

A statement that covers the needs of the blackjack application is:

Theorem. *Assume that objective function $f(x)$ and noise variables $W(x)$ are bounded in absolute value uniformly on the domain D and that*

$$E[W(x)] = 0, \quad \forall x \in D.$$

Table 1
Average performance of learning rule applied to a wiggly function

Decision time n	Average (up to n) performance
200	−1.3925
400	−1.5552
600	−1.6377
800	−1.6538
1000	−1.6677

Suppose also that the random variables $W(x(j))\,|\,x(j)$ and $W(x(n))\,|\,x(n)$ are orthogonal when $j\neq n$, and furthermore are mutually i.i.d. for all $x(j)=x(n)$, $j\neq n$. Then under the learning strategy of Section 2.2, we have that for $f_{\min}$ the essential (with respect to $p(\cdot)$) minimum of $f(\cdot)$ over D, given any positive number ε, for all resample (i.e., Steps 3, 4) times n,

$$P[f(X(n))>f_{\min}+\varepsilon]\leqslant Q\cdot \mathrm{NP}(n)\exp(-\alpha N(n))+(1-q)^{\mathrm{NP}(n)}, \tag{2.8}$$

where

$$Q=[2/(1-\exp(-\alpha))],\quad \alpha=\left(\frac{\varepsilon}{3R}\right)^2,$$

for R bounding the range of $W(x)$, and $q=P[f(T)\leqslant f_{\min}+\varepsilon]$, T being distributed according to $p(x)$. If also $(2/\alpha)\log(n)<N(n)$, $\mathrm{NP}(n)<n^{1/2}$, then almost surely, as n increases,

$$\frac{1}{n}\sum_{i=1}^{n} Y(X(i))\to f_{\min} \tag{2.9}$$

and

$$\frac{1}{n}\sum_{i=1}^{n} f(X(i))\to f_{\min}. \tag{2.10}$$

Proof. If $Z_1, Z_2, \ldots$ are independent random variables taking values in a finite interval $[a,b]$, and δ is a positive number, then from Hoeffding (1963, (2.3)), for example,

$$P\left[\left|\frac{1}{n}\sum_{i=1}^{n}(Z_i-EZ)\right|>\delta\right]\leqslant 2\,\mathrm{e}^{-n\alpha}, \tag{2.11}$$

where $\alpha=2(\delta/(b-a))^2$. From this result, with $[a,b]$ being an interval sure to contain all noise values, and after a summation of (2.11), one may conclude that for every test point $t(i)$ we have the uniform bound,

$$P\left[\sup_{j\geqslant n}|m_i(j)-f(t(i))|>\delta\right]\leqslant Q\,\mathrm{e}^{-\alpha \mathrm{NS}(i)}.$$

Here $Q=2/(1-\mathrm{e}^{-\alpha})$ and $m_i(j)$ is the sample average of observations at $t(i)$ taken up to time j. These developments assure that for our search plan, and for $N(n)$ a lower bound to the number of times any test point has been sampled by time n,

$$P\left[\max_{i\leqslant \mathrm{NP}(n)}|m_i(n)-f(t(i))|\geqslant\delta\right]\leqslant \mathrm{NP}(n)\cdot Q\,\mathrm{e}^{-\alpha N(n)}. \tag{2.12}$$

Now let $f_{\min}(n)=\min_{i\leqslant \mathrm{NP}(n)} f(t(i))$. From the definition of essential minimum, there is some positive number q, such that for T chosen with distribution $p(\cdot)$,

$$P[f(T)\leqslant f_{\min}+\delta]=q.$$

This immediately gives that

$$P[f_{\min}(n)\geqslant f_{\min}+\delta]=(1-q)^{\mathrm{NP}(n)}. \tag{2.13}$$

Let J^* denote any index $j, j \leqslant \mathrm{NP}(n)$, such that $f(t(J^*)) = f_{\min}(n)$. We have

$$f(X(n)) - f_{\min}(n) \leqslant |f(t(I^*)) - m_{I^*}| + (m_{I^*} - m_{J^*}) + |m_{J^*} - f(t(J^*))|.$$

From the definition (Step 3) of I^*, $m_{I^*} - m_{J^*} \leqslant 0$, so

$$P[f(X(n)) \geqslant f_{\min} + 3\delta] \leqslant P[f_{\min}(n) \geqslant f_{\min} + \delta] + P\left[\max_{i \leqslant \mathrm{NP}(n)} |m_i(n) - f(t(i))| \geqslant \delta\right]. \tag{2.14}$$

In view of (2.13), the first term on the right is bounded by $(1-q)^{\mathrm{NP}(n)}$ and the other term is bounded by

$$Q \cdot \mathrm{NP}(n)\, \mathrm{e}^{-\alpha N(n)},$$

by virtue of (2.12). The relation (2.14) with $\varepsilon = 3\delta$ gives relation (2.8) of the theorem.

Toward confirming (2.9) and (2.10), note that the total number of new sample and retest (i.e., Step 5) times grows as $\mathrm{O}(N(n)\mathrm{NP}(n))$ and since $Y(x)$ is hypothesized to be bounded, in view of (2.3), values at these times have no influence on the convergence issue. Let $E(j)$ represent the event:

$$E(j) = \text{“}|m_i(j) - f(t(i))| \leqslant \delta,\ 1 \leqslant i \leqslant \mathrm{NP}(j)\text{”}. \tag{2.15}$$

If $E(j)$ occurs and if $f_{\min}(j) < f_{\min} + \delta$, then by our selection strategy, we are assured that $f(x(j)) < f_{\min} + 3\delta$. Let $J(n)$ denote the set of resample times j such that $j \geqslant n$.

Then

$$P\left[\left\{\bigcap_{j \in J(n)} E(j)\right\} \cap \{f_{\min}(n) \leqslant f_{\min} + \delta\}\right] \geqslant 1 - (T_1 + T_2 + T_3). \tag{2.16}$$

If for any n, the event in (2.16) occurs, then (2.10) holds. We will show that as n becomes large, the probability converges to 1. Here the T_1 term bounds the probability that one of the m_i's, $i \leqslant \mathrm{NP}(n)$, cause an $E(j)$, $j \in J(n)$, to fail. The T_2 term bounds the probability that any test point $t(v)$ acquired after time n causes $E(j)$ to fail. Finally, T_3 bounds the probability that the objective value at the best point found so far is not within δ of the essential infimum of $f(\cdot)$. We have (in view of (2.12) and theorem assumptions about rates of $N(n)$ and $\mathrm{NP}(n)$),

$$T_1 \equiv \mathrm{NP}(n) Q \exp(-\alpha N(n)) = \mathrm{O}(\mathrm{NP}(n)/n^2) = \mathrm{o}(1).$$

Setting $V(n) = \{v\colon v > n,\ \mathrm{NP}(v-1) < \mathrm{NP}(v)\}$, we calculate that

$$T_2 \equiv \sum_{V(n)} \exp(-\alpha N(v)) = \mathrm{o}(1).$$

From (2.13),

$$T_3 \equiv (1-q)^{\mathrm{NP}(n)} = \mathrm{o}(1).$$

(By accounting for specific selection of the sequences $\{N(n)\}$ and $\{\mathrm{NP}(n)\}$, one can improve the rates for T_2 and T_3.)

If the event in (2.16) holds, we have $f(x(j)) - f_{\min} < 3\delta$ for all resample times $j > n$. Since $T_1 + T_2 + T_3 \to 0$ with increasing n, (2.10) follows.

By hypothesis, the $W(x(i))$'s are orthogonal random variables with uniformly bounded variance, so by Doob (1953, Theorem 5.2 in Chapter 4), for example, $(1/n)\sum_{i=1}^{n} W(x(i)) \to 0$, almost surely. This observation, in conjunction with (2.10) implies (2.9). □

Remark. Given a positive number ε, let us agree to say that a 'mistake' occurs at time j if $X(j)$ is chosen so that $f(X(j)) > f_{\min} + \varepsilon$. Allowing that all new sample and retest times are mistakes, one sees from (2.12) that the expected number of mistakes in the first N observations grows no faster than

$$\mathrm{NP}(n) \cdot N(n) + \sum_{j=1}^{n} P[f(X(j)) > \varepsilon].$$

To the extent that (2.8) is tight, by a convexity argument, one can see that this growth is slowest when $N(n)$ and $\mathrm{NP}(n)$ grow at the same asymptotic rate. When this rate is $O(\log(n))$, the growth is $O(\log(n)^2)$. We conjecture that this cannot be improved if D is infinite. If D has only finitely many actions, the minimal growth rate of mistakes is known to be $O(\log(n))$ (Yakowitz and Lowe (1991)).

3. Learning a winning blackjack strategy

3.1. Background for blackjack learning

Thorp (1966, Table 8.3) relates what is considered to be the first winning strategy for playing blackjack under classical rules. The strategy is determined by three matrices of numbers indexed by the pairs (T_p, U) where T_p is the Total of the values of the Player's cards and U is the value of the dealer's Up card. The functions of the three matrices are, respectively, to

(i) Tell the player when to 'double down',
(ii) Tell the player when the 'split',
(iii) Tell the player when to 'stand' and when to 'draw'.

For purposes of our learning experiment, our interest will be confined to matrix (iii). However, once matrix (iii) has been constructed, one can similarly 'learn' the entries of the other two matrices. It is presumed that the blackjack rules are as in Chapter 2 of Thorp (1966). With each pair of entries (T_p, U), the matrix (iii), which is called the *standing-number* table, gives a number, which we designate by $x^*(T_p, U)$. The *ten-count strategy* is that the player should stand only if the ratio of non-10's to 10's remaining in the deck is less than this table entry $x^*(T_p, U)$. We reproduce this table as Table 2 below. Any 10 or face card is said to be a '10 card'. Our strategy will not distinguish between soft and hard totals, and toward simplifying the experiment, our blackjack strategy will not avail itself of doubling down,

Table 2
Standing numbers for Thorp's strategy

T_p	U									ACE
	2	3	4	5	6	7	8	9	10	
12	2.0	2.1	2.2	2.4	2.3	0.0	0.0	0.0	1.1	1.0
13	2.3	2.5	2.6	3.0	2.7	0.0	0.0	0.0	1.3	1.1
14	2.7	2.9	3.3	3.7	3.4	0.0	0.0	1.1	1.6	1.2
15	3.2	3.6	4.1	4.8	4.3	0.0	0.0	1.4	1.9	1.3
16	3.9	4.5	5.3	6.5	4.6	0.0	1.2	1.7	2.2	1.4
17	∞	∞	∞	∞	∞	∞	∞	∞	∞	3.1

insurance, or splitting. Soft hands will be treated as hard, since our attention is restricted to matrix (iii). If T_p is less than 12, the player should request a hit, regardless of U. If his total is more than 17, he should stand. Thus only standing numbers for T_p between 12 and 17 are given, and at 17, one stands unless the up-card is an Ace.

The feature we prize is that with our universal learning algorithm at hand, very little problem preparation is needed, other than construction of a blackjack simulator. (This comprises the bulk of our code.)

Procedurally, we are simultaneously simulating and searching. By contrast, it is clear that Thorp (1966), Braun (1980), Griffin (1988), and Wong (1975) relied on very blackjack-specific combinatorial and exhaustive-case calculations. Our methods may not require less processing time than these others, but we are certain that less coding time and problem insight is required. The learning approach is, by its nature, a way to measure and control a black box without examining the particulars inside.

Blackjack, like many other realistic statistical decision problems, is way beyond systematic solution by nonlinear programming: Construction of the standing-number table requires finding the minimum of a 60-variable discontinuous function on the basis of noisy measurements!

3.2. Assumptions about the blackjack process

We presume that the player is gambling on a one-to-one basis against the dealer. A popular shuffling convention, and the one followed by our blackjack simulator, is that reshuffling occurs when seven cards remain in the deck, regardless of the stage of play in a game. After reshuffling, the current hand is continued.

Our purpose is to learn the ten-count standing numbers through a succession of simulated blackjack games. We apply a separate learning process to each table entry. Thus for each pair $s=(T_p, U)$, which we loosely refer to as 'state', we assign a complete set of variables such as recurrence counter $\mathrm{NS}(i;s)$ telling the number of times a standing number $t(i;s)$ for state s has been sampled, a running mean $m_i(n(s);s)$ giving the average performance at that state under the i-th standing number $t(i;s)$, and $x(n(s);s)$ is the standing number chosen at the $n=n(s)$-th visit to s.

It is presumed that a one dollar bet has been placed. Thus the possible return values are −\$1, \$0, +\$1, and +\$1.5 (for a blackjack). Since our learning algorithm is directed at minimization, the value $y(x;s)$, in the notation of (2.2), is the negative of the amount won (given the standing number was x). The variable $W(x;s)$ in (2.2) represents the randomness from hand to hand, given that $s=(T_p, U)$ has occurred. The search domain for each standing number is the interval $D=[0,7]$. For each state $s=(T_p, U)$, a learning rule as in Section 2 attempts to locate the standing number $x^*(T_p, U)$ giving the minimum expected loss, conditional on this state. The objective function for the 'learner' assigned to $s=(T_p, U)$ is $f_s(x)$, which gives the expected loss for a hand at state s, given that the standing number is x, for that state, and that optimal standing numbers are used at any successor states that may occur in that hand.

In our learning code, if the player draws and the draw takes him to a T_p value less than 18, the Y value is taken to be the sample average $(1/n(s'))\sum_{i=1}^{n(s')} Y(i;s')$ associated with the successor state $s'=(T_p', U)$. The motivation here is to take advantage of the total history of outcomes that the successor state has typically experienced. As a reviewer pointed out, this approximation, made to speed up the learning process, has the potential of degrading the algorithm since information about the ten-count is lost. Our hope is that the expected return from a given state under optimal standing number does not depend sharply on the ten-count. The correct algorithm would assign the hand outcome to $Y(s)$, regardless of intermediate states. But the increased variability of this plan for moderately long learning sessions made the approximation seem worthwhile.

There are some issues that must be considered in extending the learning theory to the blackjack application.

Proposition. *For each state s and for standing number sequence $\{x(n;s)\}$ inferred by the learning algorithm, with* NP *and N satisfying all the conditions of the theorem, we have that as $n(s)\to\infty$, almost surely, at resample times $n(s)$,*

$$f_s(x(n;s)) \to f_{\min}(s),$$

where $f_{\min}(s)$ is the optimal ten-count return conditioned on passage through s.

Proof. The two ways in which the blackjack process for a given state s differs from the learning algorithm hypotheses are that (i) the outcomes of successive visits to state s may not be statistically independent, and (ii) as the process evolves, the law governing the return Y for a given state s, and for standing number x held fixed, evolves in time. We investigate these matters separately.

(i) The outcomes of visits to a state s are independent if these visits are separated by a reshuffle. Since there is an upper bound to the number of hands before reshuffling, visits are 'm-dependent' (e.g., Stout (1974)). Such processes also obey exponential inequalities of the type (2.11). Toward showing this, let $Z(1), Z(2), \ldots$ be a bounded, m-dependent time series, each term having a mean of 0. Define sample averages,

$$M(k,i) = \sum_{j=0}^{k-1} \frac{1}{k} Z(mj+i), \quad 1 \leqslant i < m.$$

Then the summands are independent and from (2.11) directly,

$$P[|M(k,i)| > \delta] \leqslant 2\,\mathrm{e}^{-\alpha k}.$$

Let

$$M(n) = \frac{1}{m} \sum_{i \leqslant i < m} M(k,i),$$

where $k = IntegerPart(n/m)$. Then $P[|M(n)| > \delta] \leqslant 2m\mathrm{e}^{-\alpha n/m}$, or for $C = 2m$ and $\beta = \alpha/m$,

$$P[|M(n)| > \delta] \leqslant C\mathrm{e}^{-\beta n}.$$

Thus bounds of the type (2.12), as in the proof of the theorem, apply to m-dependent processes.

(ii) If a draw is requested from state s, and it leads to a new state s' in the standing number table, then the outcome Y is influenced by the standing number $x(n(s'); s')$ chosen by the learning process for the successor state s'. Since during the learning process at s', $x(n(s'); s')$ is changing, and in a nonstationary manner, the assumption of constant expected return $f(x)$ and noise distribution, in (2.2), is violated.

Toward adapting the proof of the theorem to our needs here, first note that with standing number held fixed, payoffs to states with $T_\mathrm{p} = 17$ do have stationary distributions because they have no successor states in the standing number table. Thus they fall under the purview of the theorem, and so in view of (2.9), almost surely, the averages

$$\frac{1}{n(s)} \sum_{i=1}^{n(s)} Y(i; s)$$

converge to the minimizing value $f_{\min}(s)$ for every state s with $T_\mathrm{p} = 17$.

Now proceed inductively to states with $T_\mathrm{p} = 16, 15, \ldots, 12$. States with $T_\mathrm{p} = 16$ either are terminal states (i.e., the rule requires standing), or lead to terminal states, or, if an ace is drawn, lead to states with $T_\mathrm{p} = 17$. So we need to analyze what happens, under conditions of the theorem, when the objective function is randomly changing with time (write $f(x; n)$), but is uniformly (over $x \in D$) convergent in probability, and the noises are conditionally independent, as before, but have time-varying distribution.

We extend the theorem of Section 2 to our needs by showing that it still remains true even if the model (2.2) is modified to

$$y(n) := f(x(n); n) + w(x(n); n),$$

provided $f(x;n) \to f(x)$ uniformly in x, and the W's still satisfy the theorem's zero-mean, independence, and boundedness assumptions.

Let $t(1), t(2), \ldots, t(\mathrm{NP}(n))$ denote the (Step 1) test points gathered by time n and let $V(i,n)$ be those times less than n that $t(i)$ has been sampled. Define the Cesàro sum

$$Cf(i,n) = \frac{1}{\mathrm{NS}(i)} \sum_{j \in V(i,n)} f(t(i);j), \tag{3.1}$$

and recognize it to be the expectation of m_i.

We use concepts and notation from the proof of the theorem. Define I^* as in (2.4) to be the minimizer of m_i, $1 \leqslant i \leqslant \mathrm{NP}$ and let $Cf_{\min}(n) = \min_{1 \leqslant i \leqslant \mathrm{NP}} Cf(i,n)$. Write the inequalities

$$|f_{\min}(n) - f(t(I^*))| \leqslant |f_{\min}(n) - Cf_{\min}(n)| + |Cf_{\min}(n) - m_{I^*}| + |m_{I^*} - f(t(I^*))|. \tag{3.2}$$

Let $P(n)$ be the probability that that for all $n' > n$,

$$|Cf(i,n') - f(t(i))| < \delta, \quad 1 \leqslant i \leqslant \mathrm{NP}(n'), \tag{3.3}$$

and consequently for all such n',

$$\min_i |Cf(i,n') - f_{\min}(n')| < \delta. \tag{3.4}$$

By part (i) of this proof, we also have (regardless of M)

$$P\left[\max_{1 \leqslant i \leqslant \mathrm{NP}} |m_i - Cf(i,n)| > \delta\right] \leqslant \mathrm{NP}(n) C' \mathrm{e}^{-\beta N(n)}, \tag{3.5}$$

where $C' = C/(1 - \mathrm{e}^{-\beta})$. If the event in (3.3) holds, then the first term on the right of (3.2) is bounded by δ, and if the event in (3.5) holds, the final two terms are bounded by 2δ and δ, respectively.

Combine these facts and recall (2.13) to conclude that at resample times n,

$$P[|f(x(n)) - f_{\min}| > 5\delta] \leqslant \mathrm{NP}(n) \cdot C' \mathrm{e}^{-\beta N(n)} + (1-q)^{\mathrm{NP}(n)} + P(n).$$

By hypothesis, $P(n) \downarrow 0$. This gives the analog to (2.8) of the theorem. Now the remaining steps in the proof of the theorem are applicable to proving the proposition statement. □

4. Results of the learning experiment

This section is based on Kollier (1989), which contains further programming and computational details. A learning program, centered about the algorithm of Section 2, and structured as in Section 3, was encoded. Incorporation of learning was the relatively simple matter of assigning learning variable indices for each of the 60 standing numbers, $x(T_p, U)$, $12 \leqslant T_p \leqslant 17$, $U = 2, 3, \ldots, 10$, ACE in the standing number

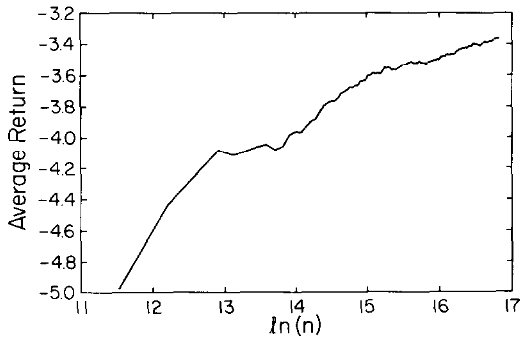

Fig. 2. Average payoff (in %) vs. number of games.

table to be learned. For each entry $s=(T_p, U)$, the domain D is the interval from 0 to 7, and at new sample times, points are drawn uniformly from D. A decision epoch for a given entry (T_p, U) occurs whenever, in a sequence of simulated games, this state is reached. Here we will report the results of a relatively mammoth learning session of 20000000 hands. The result is in keeping with shorter sessions described in Kollier (1989).

Figure 2 gives a typical average performance history during a learning session. The ordinate gives the average (over the first n games) return, in percent of bet per game. The abscissa demarks the common logarithm of the number of games played.

Table 3 gives the tens-count standing numbers learned during the 20 million-hand session. Each entry $s=(T_p, U)$ is the standing number $t(I^*, s)$ associated with the minimum $m_{I^*}(s)$ of the sample averages at the end of the session. This table bears comparison with the Braun–Thorp standing numbers of our Table 2. Some numbers are close, others are not. Plots determined by Kollier (1989) of $m_i(s)$, as a function of test point $t(i;s)$, with s fixed, and $1 \leqslant i \leqslant \mathrm{NP}(n;s)$, suggest that in many cases, the performance is not very sensitive to standing number. From Epstein (1967), Figure 7.5, for example, one sees that the ten-count ratio tends to hover between 2.0 and 2.5. In this range, our table tends to be quite accurate. Outside that range, there is

Table 3
The best standing numbers after twenty million games

T_p	U									ACE
	2	3	4	5	6	7	8	9	10	
12	2.0	2.1	1.9	2.9	2.3	1.0	0.7	0.8	0.8	0.8
13	3.6	2.4	3.7	2.9	3.7	0.4	1.5	0.7	0.2	1.2
14	2.8	3.2	4.1	3.3	4.3	0.7	1.0	0.2	1.8	1.1
15	2.6	2.9	6.2	4.3	3.1	0.6	0.6	1.5	1.7	0.6
16	4.6	6.6	5.3	4.6	4.6	1.8	0.6	2.1	2.5	1.5
17	6.8	6.8	5.7	7.0	6.2	5.1	3.9	5.1	4.9	3.9

some error, because of relative sparsity of state occurrences. But error here does less damage, since in play, these states seldom occur.

A central aim in the experiment was to ascertain whether a good strategy was, indeed, learned. The criterion of goodness is taken to be performance comparable to that achieved using Thorp's ten-count standing numbers.

Because doubling, splitting, and insurance were not included in our simulations, it is to be expected that the performance will not come up to the level of return in Thorp (1966), even if the standing numbers are learned perfectly. As benchmark for achievable performance, we simulated our reduced game using Thorp's standing numbers (our Table 2). After 5 million games, under the Thorp standing number table (but, as in the simulations, no doubling down, insurance, or splitting, and using the standing number table regardless of whether the hand is hard or soft), the average 'winning' per game was -1.592% of the amount gambled. Under the learned standing number Table 3, it was -1.70%, to the accuracy shown, after 5 million hands. This performance indicates to the authors that the experiment was reasonably successful, and the 'black-box' learning plan is feasible in this application.

References

Baldwin, R., W. Cantey, H. Maisel and J. MacDermott (1956). The optimum strategy for blackjack. *J. Amer. Statist. Assoc.* **51**, 429–439.

Bellman, R. (1978). *An Introduction to Artificial Inteligence.* Boyd and Fraser, San Francisco, CA.

Braun, J.H. (1980). *Winning Blackjack.* Data House, Chicago, IL.

Devroye, L.P. (1978a). The uniform convergence of nearest neighbor function estimators and their application in optimization. *IEEE Trans. Inform. Theory* **24**, 142–151.

Devroye, L.P. (1978b). Progressive global random search of continuous functions. *Math. Programming* **15**, 330–342.

Doob, J.L. (1953). *Stochastic Processes.* Wiley, New York.

Epstein, R.A. (1967). *The Theory of Gambling and Statistical Logic.* Academic Press, New York.

Feigenbaum, E.A. and J. Feldman, Eds. (1963). *Computers and Thought.* McGraw-Hill, New York.

Fu, K.S. (1970). Learning control systems—review and outlook. *IEEE Trans. Automat. Control* **15**, 210–221.

Griffin, P.A. (1988). *The Theory of Blackjack,* 4th ed. Huntington Press, Las Vegas, NV.

Gurin, L.S. (1966). Random search in the presence of noise. *Engineering Cybernetics* **4**, 252–260.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *J. Amer. Statist. Assoc.* **58**, 13–30.

Jarvis, R.A. (1975). Optimal strategies in adaptive control: a selective survey. *IEEE Trans. Man Machine Cybernet.* **5**, 83–94.

Kollier, M. (1989). Machine learning: an application to blackjack. Report for the Master of Science Degree, Systems and Industrial Engineering Department, University of Arizona.

Matyas, J. (1965). Random optimization. *Automat. Remote Control* **26**, 244–251.

Robbins, H. (1952). Some aspects of the sequential design of experiments. *Bull. Amer. Math. Soc.* **58**, 527–535.

Samuel, A.L. (1963). Some studies in machine learning using the game of checkers. In: E.A. Feigenbaum and J. Feldman, Eds., *Computers and Thought.* McGraw Hill, New York, 71–105.

Stout, W.F. (1974). *Almost Sure Convergence*. Academic Press, New York.

Thorp, E.O. (1966). *Beat the Dealer*, 2nd ed. Vintage, New York.

Von Neumann, J. (1958). *The Computer and the Brain*. Yale University Press, New Haven, CT.

Wiener, N. (1961). *Cybernetics*, 2nd ed. MIT Press, Cambridge, MA.

Wong, S. (1975). *Professional Blackjack*. Morrow, La Jolla, CA.

Yakowitz, S. (1989). A statistical foundation for machine learning, with application to go-moku. *Comput. Math. Appl.* **17**, 1085-1102.

Yakowitz, S. and L. Fisher (1973). On sequential search for the maximum of an unknown function. *J. Math. Anal. Appl.* **41**, 234-259.

Yakowitz, S. and E. Lugosi (1990). Random search in the presence of noise, with application to machine learning. *SIAM J. Sci. Statist. Comput.* **11**(4), 702-712.

Yakowitz, S. and W. Lowe (1991). Nonparametric bandits. *Ann. Oper. Res.* **28**, 297-312.

Yakowitz, S., T. Jawayardena and S. Li (1992a). Machine learning for dependent observations, with application to queueing network optimization. *IEEE Trans. on Automatic Control*, to appear.

Yakowitz, S., R. Hayes and J. Gani (1992b). Automatic Learning for Dynamic Markov Fields, with Application to Epidemiology. *Oper. Res.*, to appear.