TUTORIALS IN OPERATIONS RESEARCH
INFORMS 2008

© 2008 INFORMS | ISBN 978-1-877640-23-0
DOI 10.1287/educ.1080.0039

Optimal Learning

Warren B. Powell and Peter Frazier
Department of Operations Research and Financial Engineering, Princeton University, Princeton, New Jersey 08544 {powell@princeton.edu, pfrazier@princeton.edu}

Abstract Optimal learning addresses the problem of efficiently collecting information with which to make decisions. These problems arise in both offline settings (making a series of measurements, after which a decision is made) and online settings (the process of making a decision results in observations that change the distribution of belief about future observations). Optimal learning is an issue primarily in applications where observations or measurements are expensive. These include expensive simulations (where a single observation might take a day or more), laboratory sciences (testing a drug compound in a lab), and field experiments (testing a new energy saving technology in a building). This tutorial provides an introduction to this problem area, covering important dimensions of a learning problem and introducing a range of policies for collecting information.

Keywords statistical learning; Bayesian learning; stochastic optimization; dynamic programming

1. Introduction

We are surrounded by situations in which we need to make a decision or solve a problem, but where we do not know some or all of the data for the problem perfectly. Will the path recommended by my navigation system get me to my appointment on time? Will a new material make batteries last longer? Will a molecular compound help reduce a cancer tumor? If I turn my retirement fund over to this investment manager, will I be able to outperform the market? Sometimes the decisions are very simple (which investment adviser should I use), whereas others are much more complex (how do I deploy a team of security agents to assess the safety of a set of food processing plants?). Sometimes we have to learn while we are doing (what is the best path to drive to work?), whereas in other cases we may have a budget to collect information before making a final decision.

There are some decision problems that are hard even if we assume we know everything perfectly: planning the routes for aircraft and pilots, optimizing the movements of vehicles to pick up and deliver goods, or scheduling machines to finish a set of jobs on time. This is known as deterministic optimization. Then there are situations where we have to make decisions under uncertainty, but where we assume we know the probability distributions of the uncertain quantities: how do we allocate emergency trailers to respond to the next hurricane, or how do I allocate investments to minimize risk while maintaining a satisfactory return? This is known as stochastic optimization.

In this tutorial, we introduce problems where the probability distributions are unknown, but where we have the opportunity to collect new information to improve our estimates of parameters. We are primarily interested in problems where the cost of the information is considered "significant," which is to say that we are willing to spend some time thinking about how to collect the information in an effective way. What this means, however, is highly problem dependent. I am willing to spend quite a bit before I drill a $10 million hole hoping to find oil, but I may be only willing to invest a small effort before determining the next measurement inside a search algorithm running on a computer.

The modeling of learning problems, which might be described as "learning how to learn," can be particularly difficult. Although expectations are at least well defined for stochastic optimization problems, they take on subtle interpretations when we are actively changing the underlying probability distributions. For this reason, we tend to work on what might otherwise look like very simple problems. Fortunately, there is an extremely large number of these "simple problems" that would be trivial if we only knew the values of all the parameters.

2. Some Illustrations

At the risk of disguising the wide range of problems in which optimal learning arises, we are going to illustrate a few applications using some simple problems that can be viewed as stochastic network problems.

2.1. Learning the Best Path

Our first problem involves finding the best path to get from your new apartment to your new job in Manhattan. We can find a set of paths from the internet, but we do not know anything about traffic congestion or subway delays. The only way we can get data to estimate actual delays on a path is to travel the path. We wish to devise a strategy that governs how we choose paths so that we strike a balance between traveling on long paths and what we are learning.

Assume that our network is as depicted in Figure 1. Let p be a specific path, and let $x_p = 1$ if we choose to take path p. After we traverse the path, we observe a cost $\hat{c}_p$. After n trials, we can compute a sample mean $\bar{\theta}_p^n$ of the cost of traversing path p, and a sample variance $\hat{\sigma}_p^{2,n}$ using our observations of path p. Of course, we only observe path p if $x_p^n = 1$, so we might compute these statistics using

$$N_p^n = \sum_{n'=1}^{n} x_p^{n'}, \tag{1}$$

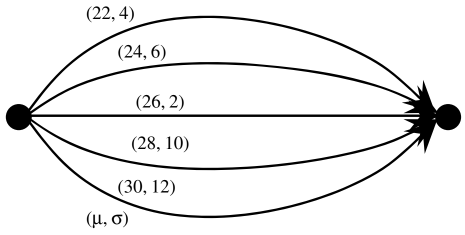
$$\bar{\theta}_p^n = \frac{1}{N_p^n} \sum_{n'=1}^{n} x_p^{n'} \hat{c}_p^{n'}, \tag{2}$$

$$\hat{\sigma}_p^{2,n} = \frac{1}{N_p^n - 1} \sum_{n'=1}^{n} x_p^{n'} (\hat{c}_p^{n'} - \bar{\theta}_p^n)^2. \tag{3}$$

Note that $\hat{\sigma}_p^{2,n}$ is our estimate of the variance of $\hat{c}_p$ by iteration n (assuming we have visited path p $N_p^n > 1$ times). The variance of our estimate of the mean, $\bar{\theta}_p^n$, is given by

$$\bar{\sigma}_p^{2,n} = \frac{1}{N_p^n} \hat{\sigma}_p^{2,n}.$$

FIGURE 1. A simple path problem, giving the current estimate of the mean and standard deviation (of the estimate) for each path.



Now we face the challenge: which path should we try? Let's start by assuming that you just started a new job, you have been to the Internet to find different paths, but you have not tried any of them. If your job involves commuting from a New Jersey suburb into Manhattan, you have a mixture of options that include driving (various routes) and commuter train, with different transit options once you arrive in Manhattan. But you do have an idea of the length of each path, and you may have heard some stories about delays through the tunnel into Manhattan, and a few stories about delayed trains. From this, you construct a rough estimate of the travel time on each path, and we are going to assume that you have at least a rough idea of how far off these estimates may be. We denote these initial estimates using

$\bar{\theta}^0_p =$ initial estimate of the expected travel time on path p;
$\bar{\sigma}^0_p =$ initial estimate of the standard deviation of $\bar{\theta}^0_p$.

We have to emphasize that $\bar{\sigma}^0_p$ is the standard deviation describing our uncertainty in our estimate $\bar{\theta}^0_p$. If we believe that our estimation errors are normally distributed, then we think that the true mean, μ_p, is in the interval $(\mu_p - z_{\alpha/2}\bar{\sigma}^0_p, \mu_p + z_{\alpha/2}\bar{\sigma}^0_p)$ $\alpha\%$ of the time. If we assume that our errors are normally distributed, we would say that we have an estimate of μ_p that is normally distributed with parameters $(\bar{\theta}^0_p, (\bar{\sigma}^0_p)^2)$.

So which path do you try first? If our priors are as shown in Figure 1, presumably we would go with the first path because it has a mean path time of 22 minutes, which is less than any of the other paths. But our standard deviation around this belief is 4, which means we believe this could possibly be as high as 30. At the same time, there are paths with times of 28 and 30 with standard deviations of 10 and 12. This means that we believe that these paths could have times that are even smaller than 20. Do we always go with the path that we think is the shortest? Or do we try paths that we think are longer, but where we are willing to acknowledge that we just are not sure, and which might be better?

If we choose a path we think is best, we say that we are *exploiting* the information we have. If we try a path because it might be better, which would help us make better decisions in the future, we say that we are *exploring.* Balancing the desire to explore versus exploit is referred to in some communities as the *exploration versus exploitation* problem. Another name is the *learn versus earn* problem.

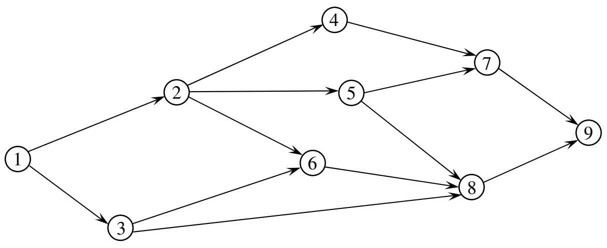
2.2. Designing a System

Our shortest path problem in the previous section is an instance of what is known as an *online* learning problem. That is, we incur the cost of traveling a link as we are learning about the cost. In our terminology, online learning refers to problems where measurements occur as we operate the system, which we might operate over a finite or infinite horizon. Offline learning, on the other hand, implies a finite learning budget, after which we incur a different cost (or reward) for operating the system with the best design we have found. Both types of learning are sequential.

There are many problems where we have to take a series of steps to design a device or process. Once we have finalized our design, we have to live with the design. Perhaps we want to find the best layout for a factory, or how to design a compound to fight cancer, or we may want to evaluate technologies for homeland security applications. In each case, we have to run experiments or computer simulations, trying out different designs and strategies. We have a budget which might be measured in experiments or dollars. After we have spent our design budget, we have to live with the system we have designed. Because our experiments are run before we have to incur the true cost of our design, this is known as offline learning.

Just as we need sometimes to try paths that might look bad, but which could actually be good, we have to try designs that might seem bad, but which could actually be quite good. The challenge is deciding when to spend our valuable information budget trying ideas that do not seem to be the best, but which might be.

FIGURE 2. A simple shortest path problem, giving the current estimate of the mean and standard deviation (of the estimate) for each path.



2.3. A Learning Shortest Path Problem

One of the best-known problems in operations research (and optimization) is the shortest path problem. Depicted in Figure 2, the problem is to find the best path from an origin (node 1) to the destination (node 9). If the costs on all the links are known, there are fast and very simple algorithms to find the best path that work very quickly even on extremely large networks.

The problem gets a little more interesting if we do not know the costs on the arcs. We get different flavors of the problem depending on how we assume the costs are revealed:

Type I: The cost on a link is revealed only after traversing the link.

Type II: The cost on link (i,j) is revealed when the traveler arrives at node i.

Type III: The cost on each link over the entire network is revealed before the traveler starts his trip.

For Type I networks, we solve the problem by setting the cost on each link equal to its mean, and then applying our deterministic shortest path algorithm. The same is true for Type III networks, except that this time we use the (presumed) known cost on each link, given to us before we start our trip. The more challenging problem is posed by Type II networks, where the cost on a link is revealed when the traveler arrives to the beginning of the link.

Our interest in this problem arises when we do not even know the probability distribution of the cost of each link. As with our simple network in §2.1, we might reasonably assume that we have an idea of the cost of the link, which is to say that we have a prior belief. But as we traverse a link, we observe the cost over the link and then update our estimates of the mean and variance. Before we start any trips, we might represent our belief state by $K^0 = (\bar{\theta}^0, (\bar{\sigma}^0)^2)$, where $\bar{\theta}^0$ is the vector of expected costs for all the links, and $(\bar{\sigma}^0)^2$ is the vector of variances. Each time we choose a path, we learn the cost over each link we traverse.

This problem is similar to the simple network in Figure 1, with one critical difference. As the traveler moves through the network, his state is captured not only by his state of knowledge (which evolves as he observes the cost on each link), but also his physical state, which is the node at which he is located. Let R^n be the node at which he is located (we use "R" to represent the state of our "resource"), where n captures the number of link transitions. Let $K^n = (\bar{\theta}^n, (\bar{\sigma}^n)^2)$ be our state of knowledge, where as before, our knowledge of a link does not change unless we traverse that link.

3. Stochastic Optimization vs. Optimal Learning

It is easy to confuse the challenges of stochastic optimization and optimal learning. Consider a classical stochastic optimization problem known as the newsvendor problem. We wish to order a quantity (of newspapers, oil, money, windmills) x to satisfy a random demand D (that is, D is not known when we have to choose x). We earn p dollars per unit of satisfied demand, which is to say $\min(x, D)$, and we have to pay c dollars per unit of x that we order. The total profit is given by

$$F(x, D) = p\min(x, D) - cx.$$

The optimization problem is to solve

$$\min_x \mathbb{E}F(x, D).$$

There are a number of ways to solve stochastic optimization problems such as this. One of the simplest is a stochastic gradient algorithm which looks like

$$x^n = x^{n-1} - \alpha_{n-1}\nabla F(x^{n-1}, D^n), \tag{4}$$

where x^{n-1} was our previous solution (computed in iteration $n-1$), and D^n is a random observation of D made at iteration n. α_{n-1} is a positive step size.

There is an elegant theory that shows under certain (fairly general) conditions that the sequence x^n will, in the limit, converge to the optimal solution. But there is no effort to use any observations to improve our distribution of belief about the stochastic process driving the problem. A central feature of optimal learning problems is the ability to make decisions that reveal information, which is then used to reduce the overall level of uncertainty about the system. Of course, our goal is to make these measurement decisions as efficiently as possible.

There is a vast array of stochastic optimization problems which involves finding the best decision (for single-stage problems) or the best policy (for multistage problems) that optimizes the expected value of some performance measure. These problems have to be solved in the presence of some form of uncertainty about one or more parameters (demands, prices, travel times) which are unknown, but where the probability distributions are assumed known. Our interest is with problems where these distributions are essentially unknown (although we often assume we have an initial distribution of belief about the parameter). We can make measurements to update our distribution of belief. There are many problems where these measurements are time consuming and/or expensive, and as a result we want to make them carefully. Optimal learning arises when we want to explicitly optimize how these measurements are made.

4. Elements of a Learning Problem

Section 2 illustrates just a few of the dimensions that arise in a learning problem. In this section, we address all the dimensions of a stochastic, dynamic problem to try to expose the range of problem classes. Unfortunately, it is unlikely that one method will be found that will solve all problem classes, so this discussion helps to expose the scope of our problem.

4.1. The States of Our System

For our purposes, we are interested in distinguishing three major problem classes:

Class I: Pure physical state. The state of knowledge about our system does not change over time.

Class II: Pure knowledge state. This is a pure learning problem, as we encountered in §§2.1 and 2.2.

Class III: Physical and knowledge states. This is what we encountered in §2.3.

We use the notation R_t to represent the physical state of the system (the variable R derives from the fact that we are typically managing "resources," and we let K_t represent our state of knowledge). The state of knowledge only arises when we have unknown parameters, and where we want to make decisions that influence how well we know these parameters. In this case, K_t captures everything we know about these parameters.

There are generally two interpretations of K_t. The first is the Bayesian view, where K_t captures the probability distribution representing our belief about the parameters. This view requires that we start with some sort of initial distribution (the "prior") before we have collected any data. The second perspective is called the frequentist view, which captures our state of knowledge through a set of statistics computed from observations. These statistics are typically *sufficient statistics*, which is to say that they capture everything we need to know from history. For example, if we are working with normally distributed data, we only need to keep the mean, variance, and number of observations. In more complex situations, we might have to retain the entire history of observations.

We note that even simple problems with a pure knowledge state remain computationally intractable, especially for many of the variations we describe below. But despite their inherent complexity, there are also many problems that involve a physical state as well as a knowledge state. One example was given in §2.3. A simpler example might be a mobile sensor for detecting nuclear radiation. Our ability to collect information depends on the location of the sensor, and it takes time and money to change the location.

4.2. Types of Decisions

The complexity of a problem depends in large part on the nature of the decision we have to make to make a measurement. Major problem classes include

(1) Binary decisions—We can continue to collection information, or stop; we can decide to show a document to an expert, or not.

(2) Discrete choice—Here, we have a set of discrete choices (not too large—dozens, hundreds, perhaps thousands), where at each point in time we have to choose one of these choices to explore. A discrete choice could be a person to do a job, a technology, a drug compound, or a path through a network.

(3) A discrete vector—We have to choose a K-dimensional vector x (perhaps of 0s and 1s). For example, out of K research proposals, we have to choose which to fund.

(4) A scalar, continuous variable—We have to choose a quantity, price, location of a facility, or concentration of a chemical that we need to optimize.

(5) A continuous vector—We have to choose $x \in \Re^K$, as we might in a linear or nonlinear program.

We will let x represent a generic "decision." We might have $x \in (0,1)$, or $x = (1,2,\ldots,M)$, or $x = (x_d)_{d\in\mathcal{D}}$ where $d \in \mathcal{D}$ is a type of decision ("fly to Chicago," "try a particular drug compound") where x_d can be binary, discrete, or continuous. We will let $\mathcal{X}$ be the set of feasible decisions. It is very important from a computational perspective to understand the nature of x, because there are problems where we assume that we can easily enumerate the elements of $\mathcal{X}$.

4.3. Exogenous Information

Exogenous information clearly comes in a wide variety depending on the application. We are primarily interested in our ability to generalize what we learn from an observation. The simplest case to analyze is where we learn nothing; the observations from measurement x tell us nothing about a different measurement x'. This is problematic when the set of potential measurements is quite large, as it often is. There are different ways in which we can generalize the results of an observation:

(1) Measurements are made of a continuous (possibly scalar) function. We might be sampling disease within a population, the response due to a particular drug dosage, or the demand response to the price of a product. Measurements are correlated inversely proportional to the distance between two measurements.

(2) Measurements of multiattribute entities. We might be predicting the importance of a document (based on the attributes of the document) or the likelihood that someone will default on a loan (as a function of an individual's financial characteristics).

(3) Estimating the time on a path in a network. The observed travel time over one path will be correlated with other paths that share common links.

(4) Effectiveness of a drug compound. Different drug compounds will share common atomic subsequences, which interact and determine the effectiveness of a drug. Drug compounds sharing common atoms (typically in specific locations) may exhibit correlations in their effectiveness.

We are going to represent exogenous information generically using the variable W. We might let W_x be the information we observe if we choose action (measurement) x, or we might let $W(S,x)$ be what we observe if we are in state S and take action x. In some instances, it is useful to let W_t be information that could be observed at time t, but we will then use x to determine which of this information we are allowed to "see." This is exactly the approach that was used in Equations (1)–(3), where $\hat{c}_p^n$ is the cost that we would have observed for path p at iteration n if we had chosen that path.

The key to handling large-scale problems (that is, where the space of possible measurements is extremely large) is to learn through some form of generalization. Fortunately, this is almost always the case for large problems. One measurement provides information about other measurements. However, there are different ways in which this type of generalization can occur, and it will be necessary to take advantage of the properties of different applications.

4.4. Transition Functions

If there is a physical state R_t, we are going to assume we are given a function that describes how this evolves over time, which we write as

$$R_{t+1} = R^M(R_t, x_t, W_{t+1}).$$

$R^M(\cdot)$ is called the "resource transition function." In traditional dynamic models (where the knowledge state is held constant), this would be the transition function, system model, or simply "the model." For our purposes, we do not need to get into the details of this function, which can vary widely.

We assume that with each decision x_t, we learn something that allows us to update our state of knowledge. We represent this generically using

$$K_{t+1} = K^M(K_t, x_t, W_{t+1}).$$

When we want to be really compact, we write the state variable as $S_t = (R_t, K_t)$ and represent its transition function using

$$S_{t+1} = S^M(S_t, x_t, W_{t+1}).$$

But we are more interested in the different ways in which we update knowledge. Earlier, we hinted at two views of the world: the frequentist view and the Bayesian view. Below, we illustrate the knowledge transition function for both views.

4.4.1. The Frequentist View. The frequentist view is arguably the approach that is most familiar to people with an introductory course in statistics. Here, we are going to compute estimates of the mean and variance of costs using the standard formulas that we first illustrated in Equations (1)–(3). This time, however, we want to give the recursive forms of these expressions.

We make a slight shift in our earlier notation, letting $\bar{\theta}_x^n$ be our estimate of the mean for decision x after n observations, and $\hat{\sigma}_x^{2,n}$ be our estimate of the variance of the observations. Finally, we let N_x^n be the number of times we have observed decision x after n iterations. We use the notational convention that we make the decision x^n using the information available at iteration n, but to be implemented for the $n+1$st iteration (before W^{n+1} becomes known). This means we begin with decision x^0, which is made before the first observation W^1 is known. These statistics can be updated recursively using (starting with $n=1$)

$$N_x^n = N_x^{n-1} + x_x^{n-1}, \tag{5}$$

$$\bar{\theta}_x^n = \begin{cases} \left(1 - \dfrac{1}{N_x^n}\right)\bar{\theta}_x^{n-1} + \dfrac{1}{N_x^n} W_x^n & \text{If } x_x^{n-1} = 1 \\ \bar{\theta}_x^{n-1} & \text{otherwise} \end{cases}, \tag{6}$$

$$\hat{\sigma}_x^{2,n} = \begin{cases} \dfrac{N_x^n - 2}{N_x^n - 1}\hat{\sigma}_x^{2,n-1} + \dfrac{1}{N_x^n}(W_x^n - \bar{\theta}_x^{n-1})^2 & \text{if } x_x^{n-1} = 1 \text{ and } N_x^n \geq 2, \\ \hat{\sigma}_x^{2,n-1} & \text{if } x_x^{n-1} = 0 \end{cases}. \tag{7}$$

Our state of knowledge is given by

$$K_{freq}^n = \left(\bar{\theta}_x^n, \hat{\sigma}_x^{2,n}, N_x^n\right)_{x \in \mathcal{X}}.$$

Equations (5)–(7) constitute a form of the knowledge transition function $K^M(\cdot)$. We emphasize that $\hat{\sigma}_x^{2,n}$ is an estimate of the variance of W_x. Typically, we are more interested in the variance of $\bar{\theta}_x^n$, which is calculated using

$$\bar{\sigma}_x^{2,n} = \frac{1}{N_x^n}\hat{\sigma}_x^{2,n}.$$

In the frequentist view, we begin with no information about the state of the system. After n observations, we have estimates of the mean of the observations W_x for each decision x, and we have an estimate of the variance of our estimate $\bar{\theta}_x^n$, given by $\bar{\sigma}_x^{2,n}$, which is based on the noise in our observations W_x^n. We would say that our knowledge of the mean and variance is based on the frequency of observations from the data. In effect, we are inferring the degree to which the estimate of the mean, $\bar{\theta}_x^n$, would bounce around if we were to repeat the experiment many times, allowing us to build up a frequency histogram of $\bar{\theta}_x^n$ over the experiments.

4.4.2. The Bayesian View. The Bayesian perspective casts a different interpretation on the statistics we compute, which is particularly useful in the context of optimal learning. In the frequentist perspective, we do not start with any knowledge about the system before we have collected any data. It is easy to verify from Equations (6) and (7) that we never use $\bar{\theta}^0$ or $\hat{\sigma}^{2,0}$. By contrast, in the Bayesian perspective we assume that we begin with a prior distribution of belief about the unknown parameters. So if μ is the true but unknown vector of means for each of the possible decisions, we might say that, although we do not know what these means are, we think they are normally distributed around μ^0 with standard deviation σ^0. In this example, we are assuming that our initial distribution of belief is normally distributed, in addition to the assumption that we know the mean and variance. This distributional assumption is known as the Bayesian prior.

Bayesian analysis begins with a simple formula that everyone learns in their first probability course. Given events A and B, the basic properties of conditional probability imply

$$P(A,B) = P(A \mid B)P(B) = P(B \mid A)P(A),$$

which implies

$$P(B \mid A) = \frac{P(A \mid B)P(B)}{P(A)}.$$

This expression is famously known as Bayes theorem. In a learning setting, the event A refers to a measurement, whereas B refers to the event that a parameter (say, the mean of a distribution) takes on a particular value. $P(B)$ refers to our initial (or prior) distribution of belief about the unknown parameter before we make a measurement, and $P(B \mid A)$ is the distribution of belief about the parameter after the measurement. For this reason, $P(B \mid A)$ is known as the *posterior distribution.*

We can apply the same idea for continuous variables. We replace B with the event that $\mu = u$, and A with the event that we observed $W = w$. Let $g(u)$ be our prior distribution of belief about the mean μ, and let $g(u \mid w)$ be the posterior distribution of belief about μ given that we observed $W = w$. We then let $f(w \mid u)$ be the distribution of the random variable W if $\mu = u$. We can now write our posterior $g(u \mid w)$, which is the density of μ given that we observe $W = w$, as

$$g(u \mid w) = \frac{f(w \mid u)g(u)}{f(w)},$$

where $f(w)$ is the unconditional density of the random variable W, which we compute using

$$f(w) = \int_u f(w \mid u)g(u).$$

Equation (8) gives us the density of μ given that we have observed $W = w$.

We illustrate these calculations by assuming that our prior $g(u)$ follows the normal distribution with mean μ_0 and variance σ_0^2, given by

$$g(u) = \frac{1}{\sqrt{2\pi}\sigma_0} \exp -\frac{1}{2}\frac{(u-\mu_0)^2}{\sigma_0^2}.$$

We further assume that the observation W is also normally distributed with mean μ and variance σ^2, which is sometimes referred to as measurement or observation error. The conditional distribution $f(w \mid u)$ is

$$f(w \mid u) = \frac{1}{\sqrt{2\pi}\sigma} \exp -\frac{1}{2}\frac{(w-u)^2}{\sigma^2}.$$

We can compute $f(w)$ from $f(w \mid u)$ and $g(u)$, but it is only really necessary to find the density $g(u \mid w)$ up to a normalization constant ($f(w)$ is part of this normalization constant). For this reason, we can write

$$g(u \mid w) \propto f(w \mid u)g(u). \tag{8}$$

Using this reasoning, we can drop coefficients such as $1/\sqrt{2\pi}\sigma_0$, and write

$$\begin{aligned} g(u \mid w) &\propto \left(\exp -\frac{1}{2}\frac{(w-u)^2}{\sigma^2}\right)\left(\exp -\frac{1}{2}\frac{(u-\mu_0)^2}{\sigma_0^2}\right), \\ &\propto \exp -\frac{1}{2}\left(\frac{(w-u)^2}{\sigma^2} + \frac{(u-\mu_0)^2}{\sigma_0^2}\right). \end{aligned} \tag{9}$$

After some algebra, we find that

$$g(u \mid w) \propto \exp -\tfrac{1}{2}\beta_1(u-\mu_1)^2, \tag{10}$$

where

$$\mu_1 = \frac{(\alpha w + \beta_0 \mu_0)}{\alpha + \beta_0}, \tag{11}$$

$$\beta_1 = \alpha + \beta_0. \tag{12}$$

The next step is to find the normalization constant (call it K), which we do by solving

$$K \int_u g(u \mid w)\, du = 1.$$

We could find the normalization constant by solving the integral and picking K so that $g(u \mid w)$ integrates to 1, but there is an easier way. What we are going to do is look around for a known probability density function with the same structure as (10), and then simply use its normalization constant. It is fairly easy to see that (10) corresponds to a normal distribution, which means that the normalization constant $K = \sqrt{\beta_1/2\pi}$. This means that our posterior density is given by

$$g(u \mid w) = \sqrt{\frac{\beta_1}{2\pi}} \exp -\frac{1}{2}\beta_1 (u - \mu_1)^2. \tag{13}$$

From Equation (13), we see that the posterior density $g(u \mid w)$ is also normally distributed with mean μ_1 given by (11), and precision β_1 given by (12) (it is only now that we see our choice of notation μ_1 and β_1 in Equations (11) and (12) was not an accident). This means that as long as we are willing to stay with our assumption of normality, that we need only to carry the mean and variance (or precision). The implication is that we can write our knowledge state as $K^n = (\mu_n, \beta_n)$ (or $K^n = (\mu_n, \sigma_n^2)$), and that (11)–(12) is our knowledge transition function.

Our derivation above was conducted in the context of the normal distribution, but we followed certain steps that can be applied to other distributions. These include the following:

(1) We have to be given the prior $g(u)$ and the conditional measurement distribution $f(w \mid u)$.

(2) We use Equation (8) to find the posterior up to the constant of proportionality, as we did in (9) for the normal distribution.

(3) We then manipulate the result in the hope of finding a posterior distribution, recognizing that we can discard terms that do not depend on u (these are absorbed into the normalization constant). If we are lucky, we will find that we have a conjugate family, and that we end up with the same class of distribution we started with for the prior. Otherwise, we are looking for a familiar distribution so that we do not have to compute the normalization constant ourselves.

(4) We identify the transition equations that relate the parameters of the posterior to the parameters of the prior and the measurement distribution (as we did with Equations (11)–(12)).

There are only a few special cases where the prior and the posterior have the same distribution. When this is the case, we say that the distribution is a *conjugate family*. The property that the posterior distribution is in the same family as the prior distribution is called *conjugacy*. The normal distribution is unusual in that the conjugate family is the same as the sampling family (the distribution of the measurement W).

In some cases, we may impose conjugacy as an approximation. For example, it might be the case that the prior distribution on μ is normal, but the distribution of the observation W is not normal (for example, it might be nonnegative). In this case, the posterior may not even have a convenient analytical form. But we might feel comfortable approximating the posterior as a normal distribution, in which case we would simply use (11)–(12) to update the mean and variance.

4.5. Objective Functions

There are two dimensions to the design of an objective function. The first dimension addresses the question, what are we trying to achieve? At the heart of optimal learning is that we are making measurements so that we can make better decisions. This means we have to have some way to evaluate "better." The second dimension addresses how we are managing the economics of measurement and the evaluation of the solution. For this dimension, we divide problems into two broad classes: online learning and offline learning.

We begin by discussing different methods for evaluating a solution.

4.5.1. Objectives. We assume that we have been collecting information using some policy π, which has put us in state S^π. If we want to identify the best policy, we have to have some measure of how well we are doing. There are three basic ways in which we do this:

- Maximizing a value or reward, or minimizing a cost or opportunity cost.
- Maximizing the likelihood of choosing the best of a set of choices.
- Finding the best fit to a set of data.

Within these three broad classes, there are different variations. We provide a list of examples of objectives in §5.

4.5.2. Designing vs. Controlling. There are two broad settings in which optimal learning arises: (1) conducting a series of measurements to design a process or a system, where we are restricted by some sort of a measurement budget; (2) controlling a system where we collect information as we are managing the system. We refer to the first case as *offline learning* because the economics of measurement are separated from the economics of using the system we have just designed. The second case is called *online learning* because we are operating the system as we are observing it. Below we briefly sketch how these objective functions might be structured.

Offline learning. There are many problems where we have a certain budget to find the best decision. There is a measurement cost $C^m(x)$ for a decision x. We let $X^\pi(S)$ represent our policy for measuring x when we are in state S. Let S^π be the state (physical state and knowledge state) that results when we use measurement policy π. We wish to find a measurement policy that solves

$$\max_\pi \mathbb{E}\left\{\max_{y\in\mathcal{Y}} C(S^\pi, y)\,\middle|\, S_0\right\}. \tag{14}$$

We assume that y represents a vector of design variables, whereas x is our measurement variables. For example, x may represent efforts to determine the degree to which different parts of the population have been exposed to a virus, and y is a vaccination plan which has to be solved subject to constraints that are captured by the feasible region $\mathcal{Y}$. Our measurements have to be made subject to a measurement budget, which we might state as

$$\sum_{n=1}^{\infty} C^m(x^n) \le B. \tag{15}$$

Of course, we assume that we stop measuring after our budget has been consumed. For many problems, $C^m(x) = 1$, and the budget B represents a limit on the number of iterations (or time periods) that we are allowed to use for measurement. $C(S^\pi, x)$ represents the optimization problem we will solve after we have completed our measurements, given our "state of knowledge" S^π, which results from our policy for collecting information.

There are important variations of this problem if we introduce risk aversion (which penalizes potentially poor solutions) or a minimax objective, where we maximize the worst case.

Online learning. Let $C(S, x)$ be a contribution (or reward, or utility) that we earn if we are in state S and choose action x. Remember that S captures our state of knowledge (in other applications, S would also include the physical state of the system). We make a decision x^n based on our state of knowledge (S^n), where we receive a reward $C(S^n, x^n)$. Let $X^\pi(S)$ be the policy we use to choose the action x. We would like to find a policy to solve

$$\max_{\pi \in \Pi} \mathbb{E}\left\{\sum_{n=0}^{N} C(S^n, X^\pi(S^n))\right\}. \tag{16}$$

Because we are solving the problem over multiple time periods, we have an incentive to explore certain decisions now so that we may make better decisions in the future. However, we have to pay for those decisions as they are made in the form of reduced rewards.

5. Objectives

Below we provide an indication of some objectives we might use. Perhaps not surprisingly, there is a vast range of possible objectives that we might use. This discussion simply hints at the diversity of perspectives.

5.1. Expected Value

Assume we have a set of choices $x \in \mathcal{X}$ (paths in a network, component designs, cancer drugs). If we take a frequentist view, our estimate of the value of choice x after one or more measurements is given by $\bar{\theta}^\pi_x$. The frequentist view evaluates the quality of the solution by making using the estimates to make a decision, as in

$$x^* = \arg\max_{x \in \mathcal{X}} \bar{\theta}^\pi_x. \tag{17}$$

The quality of the decision is then evaluated by assuming a particular truth μ, giving us

$$V^\pi_{\text{freq}}(\mu) = \mathbb{E}^\pi_W \left\{\max_{x \in \mathcal{X}} \mu_x\right\}. \tag{18}$$

Here, the expectation $\mathbb{E}_W$ is over all possible measurements we might make. The best way to think of calculating this is to fix our measurement policy and the true value μ, then perform 10,000 samples of observations from following this policy, and then take an average. The only source of noise is in the randomness of the observations.

In a Bayesian view, we would follow a measurement policy π to obtain a distribution of the mean (we continue to assume it is normally distributed), which can represent as $N(\mu^\pi, (\sigma^2)^\pi)$. Here, μ^π is the same as $\bar{\theta}^\pi$ in that is our estimate of the mean based on the same information W. Instead of assuming a particular truth μ as we did with the frequentist view, the Bayesian view assumes that we start with a distribution of belief about μ^0 before we have taken any measurements. Under this framework, we would evaluate a policy using

$$V^\pi_{\text{Bayes}} = \mathbb{E}^\pi_{\mu^0, W} \left\{\max_{x \in \mathcal{X}} \mu^\pi_x\right\}. \tag{19}$$

Here, the expectation is over the prior distribution of $\mu^0 = (\mu^0_x)_{x \in \mathcal{X}}$, and the measurements W. The difference between the frequentist and Bayesian views is that the frequentist assumes a truth, whereas the Bayesian takes an expectation over a prior distribution of belief about the possible truths. This means that

$$V^\pi_{\text{Bayes}} = \mathbb{E}_{\mu^0} V^\pi_{\text{freq}}(\mu^0).$$

Regardless of whether we are using Bayesian or frequentist perspectives, we would write the problem of finding the best policy as

$$V^* = \max_\pi V^\pi.$$

Often, maximizing over a set of policies involves simply comparing two policies. For example, perhaps our current method for running experiments (call it policy π_1) is to test each $x \in \mathcal{X}$ five times and take an average (which is to say that we are using frequentist updating). We then find the best value of x by choosing the one that produces the highest value of $\bar{\theta}_x^{\pi_1}$, which means we are computing

$$V^{\pi_1} = \bar{\theta}^{\pi_1}_{x_1^\pi}.$$

This is a natural way to evaluate a policy. We could then propose another policy π_2 and evaluate it the same way, and then choose the best policy based on a comparison of V^{π_1} and V^{π_2}. Of course, we would want to compute statistical confidence intervals around these two estimates to know if they are statistically different.

It is very common to minimize the *opportunity cost*, which measures how much worse we are doing than the optimal (whether we are minimizing or maximizing). The opportunity cost is typically viewed presented as a loss function. The frequentist version of the loss function would be written

$$L^\pi_{\text{freq}\,|\,W^\pi(\omega)} = E^\pi_W\{\mu^* - \mu_{x^*}\},$$

where

$$x^* = \arg\max_{x\in\mathcal{X}} \bar{\theta}^\pi_x.$$

L^π is the conditional *loss function* given measurements $W^\pi(\omega)$ when the data is measured using policy π. The expected opportunity cost is the expectation of the conditional loss function

$$L^\pi_{\text{freq}} = \mathbb{E}^\pi_W L^\pi_{\text{freq}\,|\,W^\pi}.$$

The Bayesian L^π_{Bayes} is defined similarly.

5.2. Probability of Correct Selection

A different perspective is to focus on the probability that we have selected the best out of a set $\mathcal{X}$ alternatives. In this setting, it is typically the case that the number of alternatives is not too large, say 10 or 20, and certainly not 10,000. Assume that

$$x^* = \arg\max_{x\in\mathcal{X}} \sum_{x\in\mathcal{X}} \mu_x$$

is the best decision (for simplicity, we are going to ignore the presence of ties). If we are using a frequentist perspective, we would make the choice

$$x^n = \arg\max_{x\in\mathcal{X}} \bar{\theta}^n_x.$$

In a Bayesian framework, we would use

$$x^n = \arg\max_{x\in\mathcal{X}} \mu^n_x.$$

Either way, we have made the correct selection if $x^n = x^*$, but even the best policy cannot guarantee that we will make the best selection every time. Let $1_{\{\mathcal{E}\}} = 1$ if the event $\mathcal{E}$ is true, 0 otherwise. We write the probability of correct selection as

$$\begin{aligned} P^{CS,\pi} &= \text{probability we choose the best alternative} \\ &= \mathbb{E}^\pi 1_{\{x^n = x^*\}}, \end{aligned}$$

where the underlying probability distribution depends on our measurement policy π. The probability is computed using the appropriate distribution, depending on whether we are using Bayesian or frequentist perspectives. This may be written in the language of loss functions. We would define the loss function as

$$L^{CS,\pi} = 1_{\{x^n \neq x^*\}}.$$

Although we use $L^{CS,\pi}$ to be consistent with our other notation, this is more commonly represented as $L_{0\text{-}1}$ for "0-1 loss."

Note that we write this in terms of the negative outcome so that we wish to minimize the loss, which means that we have not found the best selection. In this case, we would write the probability of correct selection as

$$P^{CS,\pi} = 1 - \mathbb{E}^\pi L^{CS,\pi}.$$

5.3. Indifference Zone Selection

A variant of the goal of choosing the best is to maximize the likelihood that we make a choice that is almost as good as the best. Assume we are equally happy with any outcome within δ of the best. This is referred to as the *indifference zone*. Let $V^{n,\pi}$ be the value of our solution after n measurements. We require $\mathbb{P}^\pi\{\mu_{d^*} = \mu^* \mid \mu\} > 1 - \alpha$ for all μ where $\mu_{[1]} - \mu_{[2]} > \delta$, and where $\mu_{[1]}$ and $\mu_{[2]}$ represent, respectively, the best and second best choices.

We might like to maximize the likelihood that we fall within the indifference zone, which we can express using

$$P^{IZ,\pi} = \mathbb{P}^\pi(V^{n,\pi} > \mu^* - \delta).$$

As before, the probability has to be computed with the appropriate Bayesian or frequentist distribution.

5.4. Least Square Error

A different form of loss function arises when we want to fit a function to a set of data. In this setting, we think of "x" as a set of independent variables that we choose directly or indirectly. For example, we may be able to choose x directly when fitting a linear regression of the form

$$\begin{aligned} Y(x) &= \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_I x_I + \epsilon \\ &= \theta x + \epsilon, \end{aligned}$$

where Y is the observed response and ϵ is the random error explaining differences between the linear model and the responses. We choose it indirectly when our regression is in the form of basis functions, as in

$$Y(x) = \sum_{f \in \mathcal{F}} \theta_f \phi_f(x) + \epsilon.$$

Classical linear regression assumes that we are given a set of observations, which we use to fit a model by choosing θ. Let

$$Y^{n+1} = \theta x^n + \epsilon^{n+1},$$

where θ is the true set of parameters. Our indexing reflects our requirement that x^n be chosen before we observe ϵ^{n+1}. Our measure of performance is given by

$$F(Y^{(N+1)}, x^{(N)} \mid \bar{\theta}) = \sum_{n=1}^{N} (Y^{n+1} - \bar{\theta} x^n)^2,$$

which is the sample sum of squares given measurements $x^{(N)} = (x^0, \ldots, x^N)$ and observations $Y^{(N+1)} = (Y^1, \ldots, Y^{N+1})$. Ordinary least squares regression fits a model by finding

$$\bar{\theta}^{N+1} = \arg\min_{\bar{\theta}} F(Y^{(N+1)}, x^{(N)} \mid \bar{\theta}).$$

Let $F^*(Y^{(N+1)}, x^{(N)}) = F(Y^{(N+1)}, x^{(N)} \mid \bar{\theta}^{N+1})$ be the optimal solution given $Y^{(N+1)}$ and $x^{(N)}$. Sequential estimation starts with $\bar{\theta}^n$, then measures x^n, and finally observes Y^{n+1} from which we compute $\bar{\theta}^{n+1}$. This can be done easily using recursive least mean squares (which is a special case of the Kalman filter), given by

$$\bar{\theta}^{n+1} = \bar{\theta}^n - H^n x^n (\bar{\theta}^n x^n - Y^{n+1}),$$

where H^n is an $I \times I$ scaling matrix that is computed recursively (see Powell [26], §7.3).

Our focus is on choosing the measurements x^n. Classical experimental design assumes that we choose $(x^n)_{n=0}^N$ first and then fit the model. This is sometimes referred to as batch design because the entire sample is chosen first. This is equivalent to solving

$$\min_{x^{(N)}} \mathbb{E} F^*(Y^{(N+1)}, x^{(N)}),$$

where the expectation is over the random variables in $Y^{(N+1)}$.

Our interest is in sequential design, where we choose x^n given our state S^n, which includes $\bar{\theta}^n$ and the information we need to update $\bar{\theta}^n$. In a sequential learning problem, we have to use some basis for determining how well we have done. In our optimization problems, we want to maximize our expected contribution. This optimization problem determines the values of x that are most interesting. In the area of adaptive estimation, we have to specify the values of x that are most likely to be interesting to us, which we designate by a density $h(x)$, which has to be specified. In an offline learning environment, we want to choose $x^1, \ldots, x^n, \ldots, x^N$ according to a policy π to solve

$$\min_{\pi} \mathbb{E} \int_x (Y(x) - \bar{\theta}^{\pi} x)^2 h(x)\, dx,$$

where $Y(x)$ is the random variable we observe given x, and where $\bar{\theta}^{\pi}$ is the value of $\bar{\theta}$ produced when we select x^n according to π, and when we estimate $\bar{\theta}$ optimally.

This formulation requires that we specify the domain that interests us most through the density $h(x)$. An illustration of the density function arises when we are trying to sample nuclear material over a border or in a region. For such cases, $h(x)$ might be the uniform density over the region in question. When we solve online and offline optimization problems, we do not have to specify $h(x)$ explicitly. The optimization problem (e.g. (16)) determines the region within $\mathcal{X}$ that is of greatest interest.

5.5. Entropy Minimization

Entropy is a measure of uncertainty that can be used for numeric and nonnumeric data. Imagine that we are trying to estimate a parameter μ that we know with uncertainty. If our distribution of belief about μ is continuous with density $f(u)$, a measure of the uncertainty with which we know μ is given by the entropy of $f(u)$, given by

$$H(\mu) = -\int_u f_u \log(f_u)\, du.$$

The logarithm is typically taken with base 2, but for our purposes, the natural log is fine. The entropy is largest when the density is closest to the uniform distribution. If the entropy is zero, then we know μ perfectly. Thus, we can try to take measurements that reduce the entropy of the distribution that describes our knowledge about a parameter.

6. Measurement Policies

Central to the concept of optimal learning is a measurement policy. This is a rule that tells us which action x we should take next in order to observe something new. In addition, we may also be receiving rewards or incurring costs, which have to be balanced against the value of the information being gained.

6.1. Deterministic vs. Sequential Policies

Before we begin our presentation, it is important to make a distinction between what we will call *deterministic policies* and *sequential policies*. In a deterministic policy, we decide what we are going to measure before we begin making any measurements. For example, a business may decide to perform four market research studies in different parts of the country before finalizing the pricing and advertising strategy in a full roll-out to the entire country. The decision to do four studies (and their locations) is made before we have any information from any of the studies.

Most of our interest is in sequential policies, where the decision of what to measure next may depend on past measurements. For example, when we are trying to find the shortest path, we may decide to continue sampling a path if it remains competitive, or give up on a path if the observed travel times are simply too long.

6.2. Optimal Sequential Policies

We begin by providing a framework for at least conceptualizing an optimal measurement policy. As before, we let S^n be the state of our system (physical state and knowledge state) after n measurements. The transition equations are as described in §4.4. Let $V(S^n)$ be the value of being in state S^n, and let $S^{n+1} = S^M(S^n, x^n, W^{n+1})$ be the next state if we choose x^n (which may change both the physical state and the knowledge state). $C(S^n, x)$ captures our total reward, minus any measurement costs, from being in state S^n and taking action x. Assume we wish to maximize the total discount reward, with discount factor γ. Bellman's equation characterizes the optimal action using

$$V(S^n) = \max_x \big(C(S^n, x) + \gamma \mathbb{E}\{V(S^{n+1}) \mid S^n\}\big). \tag{20}$$

We let x^n represent the optimal solution to (20). We let $X^*(S)$ be the complete mapping of states $S \in \mathcal{S}$ to actions $x \in \mathcal{X}$, where $\mathcal{X}$ describes the set of feasible actions. We refer to the function $X^*(S)$ as the optimal policy if it describes the solution to (20) for all states $S^n \in \mathcal{S}$.

It may be mathematically comforting to characterize the optimal policy, but Equation (20) is virtually impossible to solve, even for very small problems. Even if the physical state is simple (or nonexistent), the simplest knowledge state uses at least one continuous variable for each action x. Calculating a value function with as few as two continuous dimensions can, in practice, be quite a challenge. Needless to say, we do not have very many problems of practical significance that meets this modest criterion. Not surprisingly, the field of optimal learning consists primarily of finding shortcuts or, failing this, good heuristics.

6.3. Heuristic Policies

Our goal, ultimately, is to find the best possible policies for learning. The reality, however, is that most of the time we are happy to find good heuristics. Below are some popular methods that have been suggested for problems that are typically associated with discrete selection problems, which is to say that the set of measurement decisions is discrete and "not too large." We will assume that x is a measurement decision, and that the set of choices $\mathcal{X} = (1, 2, \ldots, M)$, where M is on the order of 10 to perhaps thousands, but probably not millions.

We will illustrate the policies assuming a Bayesian framework, although the same ideas can be adapted to a frequentist perspective. We let μ_x^n be our estimate of the value of decision x after iteration n, and we let β_x^n be the precision (inverse variance). Let W_x^n be our observation if $x^{n-1} = x$. We update the mean and the precision only if we sample x, so our updating equations would look like

$$\mu_x^n = \begin{cases} \dfrac{\beta_x^{n-1}\mu_x^{n-1} + \beta^W W_x^n}{\beta_x^{n-1} + \beta^W} & \text{if } x^{n-1} = x \\ \mu_x^{n-1} & \text{otherwise,} \end{cases} \tag{21}$$

$$\beta_x^n = \begin{cases} \beta_x^{n-1} + \beta^W & \text{if } x^{n-1} = x \\ \beta_x^{n-1} & \text{otherwise.} \end{cases} \tag{22}$$

The reader may wish to compare these equations to (11) and (12), where we derived the Bayesian updating scheme for the mean and precision.

6.3.1. Pure Exploration. A pure exploration strategy might sample a decision $x^n = x$ with probability $1/M$ (the probabilities do not have to be the same—they just have to be strictly positive). We would only use a pure exploration policy if we were focusing purely on estimating the value of each choice, as opposed to making a good economic decision. If we really are trying to find the best value of μ^x, a pure exploration strategy means that we would spend a lot of time measuring suboptimal choices.

6.3.2. Pure Exploitation. Exploitation means making the best decision given our current set of estimates (we are "exploiting" our knowledge). So, after iteration n, we would next measure

$$x^n = \arg\max_{x \in \mathcal{X}} \mu_x^n.$$

This strategy would seem to focus our energy on the options that appear to be the best. However, it is very easy to get stuck measuring choices that seem to be the best, especially when we simply had some bad luck measuring the better choices.

6.3.3. Mixed Exploration and Exploitation. A simple strategy that avoids the limits of pure exploration and pure exploitation is to use a mixed strategy, where we explore with probability ρ (known as the exploration rate) and we exploit with probability $1 - \rho$. The value of ρ has to be tuned for each application.

6.3.4. Epsilon-Greedy Exploration. The problem with a mixed exploration/exploitation strategy with fixed ρ is that the correct balancing of exploration and exploitation changes with the number of iterations. In the beginning, it is better to explore. As we build confidence in our estimates, we would prefer to exploit more. We can do this by using an exploration probability ρ^n at iteration n given by Singh et al. [30]:

$$\rho^n = \epsilon^n,$$

where ϵ^n is a declining series of numbers that has to ensure that in the limit, each measurement is chosen an infinite number of times. We do this by setting

$$\epsilon^n = c/n$$

for $0 < c < 1$. If we explore, we would choose measurement x with probability $1/|\mathcal{X}|$. This means that in the limit, the number of times we will measure x is given by

$$\sum_{n=1}^{\infty} \frac{c}{n|\mathcal{X}|} = \infty.$$

This assures us that we will estimate each measurement x perfectly, but as the measurements progress, we will spend more time measuring what we think are the best choices.

6.3.5. Boltzmann Exploration. A different strategy for balancing exploration and exploitation is known as Boltzmann exploration. With this strategy, we sample measurement x with probability ρ_x^n given by

$$\rho_x^n = \frac{\exp(\rho\mu_x^n)}{\sum_{x' \in \mathcal{X}} \exp(\rho\mu_{x'}^n)}.$$

This policy is also known as the *soft max*. If $\rho = 0$, we are going to sample each measurement with equal probability (pure exploration). As $\rho \to \infty$, we will sample the measurement with the highest value of μ^n with probability 1 (pure exploitation). In between, we explore the better options with higher probability. Furthermore, we can make ρ increase with n (which is typical), so that we explore more in the beginning, converging to a pure exploitation strategy.

A limitation of Boltzmann exploration, especially for applications with a large number of measurements, is that computing these probabilities can be fairly expensive.

6.3.6. Interval Estimation. Imagine that instead of choosing a measurement that we think is best, we will choose a measurement that we think might eventually be best if we were to take enough measurements. With this idea, we might construct a confidence interval and then value an option based on the upper limit of, say, a 95th confidence interval. Letting α be our confidence level and denoting by $z_{\alpha/2}$ the standard normal deviate leaving $\alpha/2$ in the upper tail, our upper limit would be

$$\nu_x^n = \mu_x^n + z_{\alpha/2}\bar{\sigma}_x^n,$$

where $\bar{\sigma}_x^n = 1/\sqrt{\beta_x^n}$ is our estimate of the standard deviation of μ_x^n. When we use an interval exploration policy, first introduced by Kaelbling [21], we choose the measurement x^n with the highest value of ν_x^n.

Although the interpretation as the upper limit of a confidence interval is appealing, the confidence level α carries no particular meaning. Instead, $z_{\alpha/2}$ is simply a tunable parameter. Experimental evidence reported in Kaelbling [21], as well as our own, suggest values of $z_{\alpha/2}$ around 2 or 3 works best for many applications. Our experimental work suggests that interval exploration is not only extremely easy to use, it also works quite well (assuming $z_{\alpha/2}$ has been properly tuned). But it is not provably convergent, and in fact it is known to occasionally become stuck when a good solution has received a few bad observations.

7. Stopping Problems

Stopping problems are a simple but important class of learning problems. In this problem class, information arrives over time, and we have to choose whether to view the information or stop and make a decision. Unlike the problems we describe below, in this problem class we have no control over the information that arrives to us. We only control whether we continue to view the information, or stop and make a decision.

Stopping problems represent a rich problem class. We use a classic problem from the literature known as the *secretary problem*, first introduced by Cayley [4], which involves the challenge of interviewing a sequence of candidates for a secretarial position, but it can also be applied to reviewing a series of offers for an asset (such as selling your house or car). The problem involves the tradeoff between observing candidates (which allows us to collect information) and making a decision (exploiting the information).

Assume that we have N candidates for a secretarial position. Each candidate is interviewed in sequence and assigned a score that allows us to compare him or her to other candidates. Although it may be reasonable to try to maximize the expected score that we would receive, in this case, we want to maximize the probability of hiring the best candidate out of the entire pool. We need to keep in mind that if we stop at candidate n, then we would not have even interviewed candidates $n+1,\dots,N$.

Let

$$\begin{aligned}
\omega^n &= \text{Score of the } n\text{th candidate.}\\
S^n &= \begin{cases} 1 & \text{If the score of the } n\text{th candidate is the best so far,}\\ 0 & \text{if the score of the } n\text{th candidate is not the best so far,}\\ \Delta & \text{if we have stopped already.}\end{cases}\\
\mathcal{S} &= \text{State space, given by } (0,1,\Delta)\text{, where the states 0 and 1 mean that}\\
&\quad\ \text{we are still searching, and } \Delta \text{ means we have stopped the process.}\\
\mathcal{X} &= \{0(\textit{continue}), 1(\textit{quit})\}\text{, where ``quit'' means that}\\
&\quad\ \text{we hire the last candidate interviewed.}
\end{aligned}$$

Because the decision function uses the most recent piece of information, we define our history as

$$h^n = \{\omega^1,\dots,\omega^n\}.$$

To describe the system dynamics, it is useful to define an indicator function

$$I^n(h^n) = \begin{cases} 1 & \text{if } \omega^n = \max_{1\le m\le n}\{\omega^m\}\\ 0 & \text{otherwise,}\end{cases}$$

which tells us if the last observation is the best. Our transition function can now be given by

$$S^{n+1} = \begin{cases} I^n(h^n) & \text{if } x^n = 0 \text{ and } S^n \ne \Delta\\ \Delta & \text{if } x^n = 1 \text{ or } S^n = \Delta.\end{cases}$$

To compute the one-step transition matrix, we observe that the event the $(n+1)$st applicant is the best has nothing to do with whether the nth was the best. As a result, we can write the conditional probability that $I^{n+1}(h^{n+1}) = 1$ using

$$\mathbb{P}[I^{n+1}(h^{n+1}) = 1 \mid I^n(h^n)] = \mathbb{P}[I^{n+1}(h^{n+1}) = 1].$$

This simplifies the problem of finding the one-step transition matrix. By definition we have

$$\mathbb{P}[S^{n+1} = 1 \mid S^n, x^n = 0] = \mathbb{P}[I^{n+1}(h^{n+1}) = 1].$$

$I^{n+1}(h^{n+1}) = 1$ if the $(n+1)$st candidate is the best out of the first $n+1$, which clearly occurs with probability $1/(n+1)$. So

$$\begin{aligned}
\mathbb{P}(S^{n+1} = 1 \mid S^n, x^n = 0) &= \frac{1}{n+1},\\
\mathbb{P}(S^{n+1} = 0 \mid S^n, x^n = 0) &= \frac{n}{n+1}.
\end{aligned}$$

Our goal is to maximize the probability of hiring the best candidate. So, if we do not hire the last candidate, then the probability that we hired the best candidate is zero. If we hire the nth candidate, and the nth candidate is the best so far, then our reward is the probability that this candidate is the best out of all N. This probability is simply the probability that the best candidate out of all N is one of the first n, which is n/N. So, the conditional reward function is

$$C^n(S^n, x^n \mid h^n) = \begin{cases} n/N & \text{If } S^n = 1 \text{ and } x^n = 1,\\ 0 & \text{otherwise.}\end{cases}$$

With this information, we can now set up the optimality equations

$$V^n(s^n) = \max_{x^n \in \mathcal{X}} \mathbb{E}\{C^n(s^n, x^n \mid h^n) + V^{n+1}(S^{n+1}) \mid s^n\}.$$

It is possible to show (see Puterman [27]) that if we have N candidates to interview, that we should interview m before offering anyone the job. The first m candidates, then, are used to learn something about the quality of the candidates. After interviewing m candidates, then the best strategy is to take the first candidate who is the best candidate interviewed so far (including the first m). It is possible, of course, that in the remaining $N-m$ candidates, we never find anyone as good as the best of the first m. In this case, we simply hire the last candidate interviewed.

It turns out that the optimal value of m satisfies $\ln(N/m) = 1$ or $N/m = e$ or $m/N = e^{-1} = 0.368$. So, for large N, we want to interview 37% of the candidates, and then choose the first candidate that is the best to date.

The secretary problem is an example of a deterministic policy. Our decision of how much information to collect can be solved before we have observed any of the candidates.

8. Multiarmed Bandit Problems

One of the best known problems in information collection is known as the multiarmed bandit problem. The story that motivates the problem trivializes its importance. Assume you are facing the problem of playing M slot machines. Now pretend that if you play slot machine m at iteration n, you receive random winnings W_m^n. Further assume that the expected winnings for machine m, given by μ_m, is unknown and different for each machine. We would like to estimate μ_m, but the only way we can do this is by trying the slot machine and collecting a random observation of W_m. The problem is identical to our little transportation problem that we first introduced in §2.1, but the problem is known formally in the literature as the multiarmed bandit problem.

The basic idea is to choose an arm m (which means $x_m^n = 1$), after which we observe the winnings W_m^{n+1}. We update our estimate of the mean using either a frequentist framework (Equations (5)–(7)) or a Bayesian framework (Equations (21)–(22)).

In §6.2, we presented a dynamic program that provided a framework for solving any measurement problem optimally. We also argued that it is virtually impossible to solve optimally, and that we generally look for shortcuts and good heuristics. The multiarmed bandit problem is one of those problems that can be solved optimally, using a clever shortcut credited to J.C. Gittins (Gittins and Jones [17], Gittins [15, 16]). Gittins found that instead of solving the dynamic program with the multidimensional state variable, it was possible to characterize the optimal solution using something known as an "index policy." This works by computing an index ν_m^n for each option m at each iteration n. The index ν_m^n is computed by solving M one-dimensional problems. The index policy works by finding $m^* = \arg\max_m \nu_m^n$, which is to say the machine m^* that corresponds to the largest index. Thus, instead of solving a single M-dimensional problem, we have to solve M one-dimensional problems.

8.1. Basic Theory of Gittins Indices

The idea behind Gittins indices works as follows. Assume that we are playing a single slot machine, and that we have the choice of continuing to play the slot machine or stopping and switching to a process that pays a reward ρ. If we choose not to play, we receive ρ, and then find ourselves in the same state (because we did not collect any new information). If we choose to play, we earn a random amount W, plus we earn $\mathbb{E}\{V(S^{n+1}, \rho) \mid S^n\}$, where S^{n+1} represents our new state of knowledge resulting from our observed winnings. For reasons that will become clear shortly, we write the value function as a function of the state S^{n+1} and the stopping reward ρ.

The value of being in state S^n, then, can be written as

$$V(S^n,\rho) = \max\big[\rho + \gamma V(S^n,\rho), \mathbb{E}\{W^{n+1} + \gamma V(S^{n+1},\rho) \mid S^n\}\big]. \tag{23}$$

The first choice represents the decision to receive the fixed reward ρ, whereas in the second we get to observe W^{n+1} (which is random when we make the decision). When we have to choose x^n, we will use the expected value of our return if we continue playing, which is computed using our current state of knowledge. So, $\mathbb{E}\{W^{n+1} \mid S^n\} = \bar{\theta}^n$, which is our estimate of the mean of W given what we know after the first n measurements.

If we choose to stop playing at iteration n, then S^n does not change, which means we earn ρ and face the identical problem again for our next play. This means that once we decide to stop playing, we will never play again, and we will continue to receive ρ (discounted) from now on. The infinite horizon, discounted value of this reward is $\rho/(1-\gamma)$. This means that we can rewrite our optimality recursion as

$$V(S^n,\rho) = \max\left[\frac{\rho}{1-\gamma}, \mathbb{E}\{W^{n+1} + \gamma V(S^{n+1},\rho) \mid S^n\}\right]. \tag{24}$$

Here is where we encounter the magic of Gittins indices. Let ν be the value of ρ that makes us indifferent between stopping and accepting the reward ρ (forever), versus continuing to play the slot machine. That is, we wish to find ν that satisfies

$$\frac{\nu}{1-\gamma} = \mathbb{E}\{W^{n+1} + \gamma V(S^{n+1},\rho) \mid S^n\}. \tag{25}$$

ν depends on the state S. If we use a Bayesian perspective and assume normally distributed rewards, we would use $S^n = (\mu^n, \sigma^{2,n})$ to capture our distribution of belief about the true mean μ. If we use a frequentist perspective, we would let our estimate of the means be the vector $\bar{\theta}^n = (\bar{\theta}^n_m)_{m=1}^M$, our estimate of the variances of W would be the vector $\sigma^{2,n} = (\sigma^{2,n}_m)_{m=1}^M$, and the number of observations for each machine would be given by N^n_m. This means our state variable is $S^n = (\bar{\theta}^n, \sigma^{2,n}, N^n)$. We would find the Gittins index ν for each machine, which means we would represent the index as $\nu^n_m = \nu_m(S^n_m)$. Gittins showed that the optimal policy is to play the slot machine with the highest value of ν^n_m.

8.2. Gittins Indices for Normally Distributed Rewards

Now assume that the rewards are normally distributed with known mean μ and variance σ^2. Gittins showed (Gittins [16]) that the indices could be computed using

$$\nu_m(\bar{\theta}^n, \sigma^{2,n}, n) = \bar{\theta}^n + \Gamma(n)\sigma^n, \tag{26}$$

where $\Gamma(n) = \nu(0,1)$ is the Gittins index if the mean is 0 and the variance is 1 (this is sort of a "standard normal" Gittins index). So, although computing Gittins indices is a pain, we only have to do it for a single mean and variance and then use (26) to translate it to other distributions (this only works for normally distributed rewards). Table 1 provides $\Gamma(n)$ for both known and unknown variance cases, for discount factors of 0.95 and 0.99.

8.3. Approximating Gittins Indices

Finding Gittins indices is somewhat like finding the cumulative standard normal distribution. It cannot be done analytically, and requires instead a fairly tedious numerical calculation. We take for granted the existence of nice functions built into most programming languages for computing the cumulative standard normal distribution, for which extremely accurate polynomial approximations are available. In Excel, this is available using the function *NORMINV*.

Table 1. Gittins indices $\Gamma(n)$ for the case of observations that are normally distributed with mean 0 and variance 1 (from Gittins [16]).

	Discount factor			
	Known variance		Unknown variance	
Observations	0.95	0.99	0.95	0.99
1	0.9956	1.5758	—	—
2	0.6343	1.0415	10.1410	39.3343
3	0.4781	0.8061	1.1656	3.1020
4	0.3878	0.6677	0.6193	1.3428
5	0.3281	0.5747	0.4478	0.9052
6	0.2853	0.5072	0.3590	0.7054
7	0.2528	0.4554	0.3035	0.5901
8	0.2274	0.4144	0.2645	0.5123
9	0.2069	0.3808	0.2353	0.4556
10	0.1899	0.3528	0.2123	0.4119
20	0.1058	0.2094	0.1109	0.2230
30	0.0739	0.1520	0.0761	0.1579
40	0.0570	0.1202	0.0582	0.1235
50	0.0464	0.0998	0.0472	0.1019
60	0.0392	0.0855	0.0397	0.0870
70	0.0339	0.0749	0.0343	0.0760
80	0.0299	0.0667	0.0302	0.0675
90	0.0267	0.0602	0.0269	0.0608
100	0.0242	0.0549	0.0244	0.0554

Such functions do not exist for Gittins indices, but a reasonably good approximation has been developed by Brezzi and Lai [3]. We are going to use the adaptation given in Yao [31], which provides both lower and upper bounds, as well as a correction term that adjusts for the fact that the approximation given by Brezzi and Lai [3] is for continuous time problems, whereas the bandit problem is discrete time. The approximation works by first computing

$$\begin{aligned}
\beta &= \text{discrete time discount factor},\\
c &= \text{continuous time discount factor},\\
&= -\ln(\beta),\\
n &= \text{iteration},\\
s &= (nc)^{-1}.
\end{aligned}$$

Note that we use $n=1$ for the first iteration (when there are no observations), $n=2$ for the second iteration (when there is one observation), and so on. Brezzi and Lai [3] estimated the following approximation:

$$\Psi(s)=\begin{cases}\sqrt{s/2} & s\le 0.2,\\ 0.49-0.11s^{-1/2} & 0.2<s\le 1,\\ 0.63-0.26s^{-1/2} & 1<s\le 5,\\ 0.77-0.58s^{-1/2} & 5<s\le 15,\\ \{2\ln s-\ln\ln s-\ln 16\pi\}^{1/2} & s>15.\end{cases}$$

Yao [31] then presents the following approximate lower bound for the Gittins index, given by

$$\Gamma^{LB}(n)=\left[\frac{1}{\sqrt{n}}\Psi\left(\frac{1}{nc}\right)-\frac{0.583n^{-1}}{\sqrt{1+n^{-1}}}\right]. \tag{27}$$

The second term involving the coefficient 0.583 is the correction term that adjusts for the fact that the bandit problem is discrete time, whereas the approximation by Brezzi and Lai [3] is for continuous time.

An approximate upper bound on the Gittins index is given by

$$\Gamma^{UB}(n) = \left[\frac{1}{\sqrt{n}}\sqrt{s/2} - \frac{0.583n^{-1}}{\sqrt{1+n^{-1}}}\right], \tag{28}$$

which is the same as the lower bound except that we use $\Psi(s) = \sqrt{s/2}$ for all s. A reasonable approximation for the Gittins index is then found by simply averaging the approximate upper and lower bounds, giving us

$$\Gamma(n) \approx (\Gamma^{LB}(n) + \Gamma^{UB}(n))/2.$$

For example, consider a problem where $\beta = 0.90$, which means that our reward after n iterations is discounted back by β^n. The continuous discount factor is given by $c = -\ln(\beta) = -\ln(0.10) = 0.10536$. If we have performed $n = 10$ observations, we compute the parameter $s = 1/(nc) = 1/(10 * 0.10536) = 0.94912$. We see that s is between 0.2 and 1, so $\Psi(s) = 0.37709$. We use this to compute the approximate lower bound, giving us $\Gamma^{LB} = 0.06366$. The approximate upper bound is easily computed to be 0.16226. Averaging the approximate lower and upper bounds give us an estimate of the Gittins index of $\Gamma(10) \approx 0.11296$.

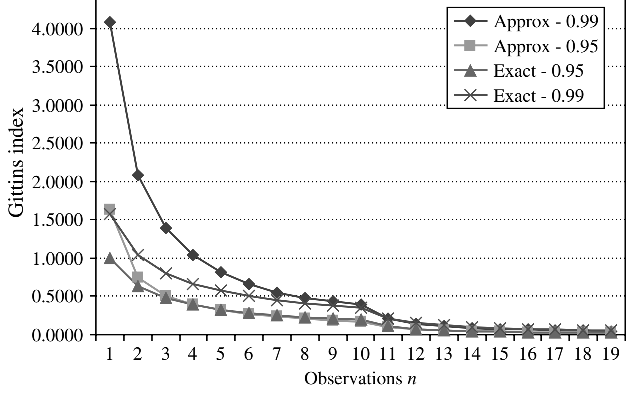
Figure 3 shows the exact and approximate Gittins indices for $\beta = 0.95$ (the lower two lines) and for $\beta = 0.99$ (the upper two lines). For both cases, the approximation is quite accurate for $n \geq 10$, and the accuracy is quite good for $\beta = 0.95$ and below. For $\beta = 0.99$, we see a fairly large error when the number of observations is less than 5, although in practice this would have little effect.

The upper and lower bounds match as $\beta \to 0$ or $n \to \infty$ because in both cases $s \to 0$, giving $\Psi(s) = \sqrt{s/2}$ (in this case, the two bounds are the same).

8.4. Remarks

The multiarmed bandit problem is an online problem, which means that we incur the costs/rewards as we are collecting information. We do not have the concept of measuring (or training), after which we use what we learned (without any further learning). Another significant characteristic of bandit problems is that it must be posed as an infinite horizon, discounted problem. Comparable results do not exist for finite horizon problems, although this does not prevent us from applying the concept heuristically.

FIGURE 3. Exact vs. approximation for Gittins indices for discount factors $\beta = 0.95$ and 0.99.



It is useful to compare Gittins indices with interval estimation. The Gittins index is computed using

$$\nu_m^{G,n} = \bar{\theta}_m^n + \Gamma(n)\sigma_m^n. \tag{29}$$

Here, σ_m^n is an estimate of the standard deviation of W_m. As $n \to \infty$, $\sigma_m^n \to \sigma_m$, the true standard deviation for W_m. By contrast, interval estimation uses the index

$$\nu_m^{IE,n} = \bar{\theta}_m^n + z_{\alpha/2}\bar{\sigma}_m^n, \tag{30}$$

where $\bar{\sigma}_m^n = \sigma_m^n/\sqrt{N_m^n}$, which is an estimate of the standard deviation of the estimate $\bar{\theta}^n$. As $n \to \infty$, $\bar{\sigma}_m^n \to 0$. The two methods should perform comparably if $\Gamma(n)$ declines roughly with $1/\sqrt{n}$. Of course, interval estimation does not provide any guidance with respect to the choice of $z_{\alpha/2}$, which remains a tunable parameter. But if we choose $z_{\alpha/2} = \Gamma(1)$, we find that $z_{\alpha/2}/\sqrt{n}$ is, in fact, a rough approximation of $\Gamma(n)$ (but not nearly as good as the approximation we provide above).

It is important to realize that using Gittins indices does not produce an algorithm that guarantees that we will eventually find the optimal solution. It is quite possible that, even in the limit, we will become stuck at a suboptimal solution. This is surprising to some, because Gittins indices are "optimal." But they are optimal in the sense of maximizing the discounted rewards, which means we care quite a bit how much we earn in the early iterations. This result should be contrasted with our "provably optimal" stochastic gradient algorithm described in §3. This algorithm guarantees that $\lim_{n\to\infty} x^n \to x^*$. However, the algorithm provides no guarantees how long it will take to reach the optimal solution (or even anything close to it). By contrast, Gittins indices focuses on getting to a good solution quickly, but there is no guarantee at all that we will eventually reach the optimal solution.

9. Ranking and Selection Problems

Ranking and selection problems are the offline version of multiarmed bandit problems. Assume that we have $m = (1, 2, \ldots, M)$ possible "designs." A design might be a type of cholesterol drug (we want the one that lowers cholesterol the most), the parameters of a chemical process (e.g. temperature, relative mix of certain inputs), or the design of a manufacturing process (choice of equipment). Testing a design might involve running a time-consuming computer simulation, or it might require a physical experiment. We assume that in either case, the measurement involves some noise, which means we may need to repeat the experiment several times. We assume that we have a budget that determines how many times we can perform these experiments, after which we have to take the best design and live with it. This means we match the model described in §4.5.2 for offline learning.

With online learning, we have to live with the rewards we receive while we are measuring. The emphasis is on rate of convergence, where we wish to maximize the discounted rewards earned. Our horizon is determined entirely by the discount factor. With offline learning, we have a design phase after which we have to live with the results. The design phase is conducted subject to some sort of budget constraint, after which we are interested in the quality of the solution. The simplest budget constraint is a limit on the number of iterations, but other applications may involve physical measurements, where the cost may even depend on what is being measured. For example, we may want to design a plan to distribute antivirals to protect against flu outbreaks, but we have to send out a team of medical technicians (e.g. nurses) to measure the spread of the disease in various regions. Travel costs can vary widely, creating incentives to collect information that is less expensive.

As with the bandit problems, the optimal policy for ranking and selection problems is characterized by the dynamic program given in §6.2. Unfortunately, we do not have a result comparable to Gittins indices to provide a practical solution. Just the same, we retain an interest in simple rules and procedures that we can use to provide guidance. This is handled through an idea known as the knowledge gradient (KG; also known as the expected value of information).

9.1. Overview of Methods

Much of the work on ranking and selection poses the problem within the *indifference zone formulation*, in which policies are required to meet a condition on the (frequentist) probability of correct selection. Recall from §5.2 that the probability of correct selection is the probability that we fail to correctly identify the best design after our sampling is complete. Understood from the frequentist viewpoint, we fix the underlying and unknown true design values when we compute this probability, rather than averaging over them according to a prior as we would in a Bayesian setting. Achieving a large probability of correct selection is more difficult on some of these design value configurations than on others and, in particular, identifying the best design is more difficult if its true value is closer to those of the other designs. With this insight in mind, we choose parameters $\alpha > 0$ and $\delta > 0$ and give the name "indifference zone" to those design value configurations for which the best design is no more than δ better than the second best. We then say that a policy meets the α, δ indifference zone criterion if this policy's probability of correct selection is greater than $1 - \alpha$ for every design configuration outside the indifference zone. This requirement may then be understood as a worst-case requirement over all the design configurations outside the indifference zone.

We will not discuss indifference zone procedures in detail here, but an excellent review of the earlier work on the indifference zone formulation may be found in Bechhofer et al. [1], and for a comprehensive discussion of later work see Bechhofer et al. [2]. Much of this work is on two-stage and multistage procedures, which decide at the beginning of each stage how many samples to take from each design in that stage. A classic algorithm of this type is the two-stage procedure in Rinott [29], and an improved two-stage procedure can be found in Nelson et al. [23]. Fully sequential procedures have also been developed in Paulson [24], Hartmann [19], Paulson [25], Kim and Nelson [22]. Among these procedures, the last also takes advantage of common random numbers to improve performance, and the use of variance reduction techniques like common random numbers to improve ranking and selection has been an important piece of modern research in ranking and selection.

Another approach to ranking and selection is the Bayesian one. Bayesian techniques developed later, although an important early discussion of single-stage procedures for Bayesian ranking and selection can be found in Raiffa and Schlaifer [28]. More recently, a number of two-stage and fully sequential Bayesian procedures have been developed. These procedures generally focus on maximizing either the Bayesian probability of correct selection or the expected mean of the chosen design. This second objective is also sometimes written equivalently as minimizing the expected linear loss. They generally assume that the samples come from a normal distribution, with either known or unknown variances.

Two main families of Bayesian procedures exist. The first, called the optimal computing budget allocation (OCBA), is a family of procedures that is usually used for maximizing the probability of correct selection. OCBA procedures choose their allocations by considering the reduction in posterior variance and the resulting improvement in probability of correct selection that a block of samples in the current stage will induce. To ease computation, they assume that the samples will not change the posterior means. They then find a sample allocation that will approximately optimize the approximate improvement. A number of OCBA procedures exist which use different approximations to arrive at an allocation, including those presented in Chen et al. [5–8], and Chen et al. [6]. Recently, an OCBA procedure for optimizing the expected value of the chosen design was presented in He et al. [20], and a procedure that uses common random numbers was presented in Fu et al. [14].

A type of procedure distinct from OCBA is the value of information procedure (VIP). Like OCBA, procedures in the VIP family estimate the expected improvement of a single stage's worth of allocations and choose that stage's allocations so as to approximately maximize this estimate. VIP differs from OCBA, however, in that its procedures consider both the reduction in posterior variance and the change in posterior mean when they estimate expected improvement. Two-stage and fully sequential VIP procedures for optimizing probability of correct selection and linear loss may be found in Chick and Inoue [10], and a two-stage procedure which also uses common random numbers is presented in Chick and Inoue [9]. Another VIP procedure, which differs from the others in that it only allocate one measurement at a time rather than in blocks of multiple measurements, is presented in Chick et al. [11]. This procedure calculates the expected improvement due a single allocation exactly, reducing the number of approximations needed. The procedure generalizes to the unknown variance case an older procedure, first introduced in Gupta and Miescke [18] as the $(R_1, \ldots, R_1)$ procedure, and analyzed more fully under the name "knowledge-gradient" procedure in Frazier et al. [13]. We will discuss this knowledge-gradient procedure more fully in the next section.

9.2. The Knowledge Gradient for Independent Measurements

We start with problems where measuring x tells you nothing about the value of $x' \neq x$. This might arise, for example, if we are evaluating different technologies for sensing nuclear radiation that have nothing in common. Or, we might be evaluating different people for managing assets, where we do not feel that we can use any characteristics about the person to predict the performance of another person.

A simple idea for solving ranking and selection problems is to use a myopic policy first introduced by Gupta and Miescke [18], and further analyzed by Frazier et al. [13] under the name *knowledge gradient*. The idea works as follows. Assume that our knowledge state at iteration n is given by $S^n = (\mu_x^n, \beta_x^n)_{x \in \mathcal{X}}$ (recall that β_x^n is the precision, which is the inverse of the variance in our estimate μ_x^n of the true mean). If we stop measuring now, we would pick the best option, which we represent by

$$V^n(S^n) = \max_{x' \in \mathcal{X}} \mu_{x'}^n.$$

Now let $S^{n+1}(x) = S^M(S^n, x^n, W^{n+1})$ be the next state if we choose to measure $x^n = x$ right now, allowing us to observe W_x^{n+1}. This allows us to update μ_x^n and β_x^n, giving us an estimate μ_x^{n+1} for the mean and β_x^{n+1} for the precision. The solution to our problem would be given by

$$V^{n+1}(S^{n+1}(x)) = \max_{x' \in \mathcal{X}} \mu_{x'}^{n+1}.$$

At iteration n, μ_x^{n+1} is a random variable. We would like to choose x at iteration n, which maximizes the expected value of $V^{n+1}(S^{n+1}(x))$. We can think of this as choosing x^n to maximizes the incremental value, given by

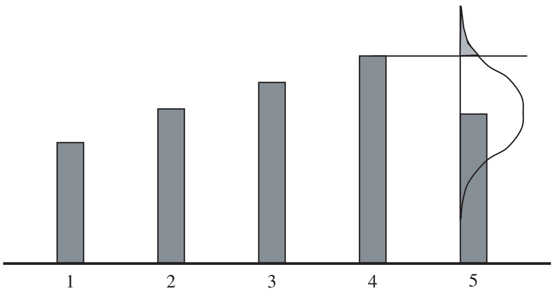
$$\nu^{KG,n} = \max_{x \in \mathcal{X}} \mathbb{E}\big[V^{n+1}(S^{n+1}(x)) - V^n(S^n) \mid S^n\big]. \tag{31}$$

We can view the right-hand side of (31) as the derivative (or gradient) of $V^n(S^n)$ with respect to the measurement x. Thus, we are choosing our measurement to maximize the gradient with respect to the knowledge gained from the measurement, hence the label "knowledge gradient," introduced by Frazier et al. [13]. This concept was first introduced by Gupta and Miescke [18] and has since been analyzed experimentally by Chick et al. [11], where it is termed the "expected value of information." We write the knowledge gradient policy using

$$X^{KG,n} = \arg\max_{x \in \mathcal{X}} \mathbb{E}\big[V^{n+1}(S^{n+1}(x)) - V^n(S^n) \mid S^n\big]. \tag{32}$$

The knowledge gradient, $\nu^{KG,n}$, is the amount by which the solution improves. This is illustrated in Figure 4, where the estimated mean of choice 4 is best, and we need to find the

FIGURE 4. Illustration of the knowledge gradient if we were to measure choice 5.



marginal value for choice 5. The estimated mean of choice 5 will move up or down according to a normal distribution (we assume with mean 0). The solid area under the curve that exceeds that for choice 4 is the probability that measuring 5 will produce a value that is better than the current best, which means that V^{n+1} will increase. The knowledge gradient is the expected amount by which it will increase (we receive a value of 0 if it does not go up).

A significant advantage of the knowledge gradient policy is that it can be quite easy to compute. Recall that the precision is simply the inverse of the variance, given by $\bar{\sigma}_x^{2,n}$. Our updating formula for the variance is given by

$$\begin{aligned}\bar{\sigma}_x^{2,n} &= \left((\bar{\sigma}_x^{2,n-1})^{-1} + \sigma^{-2}\right)^{-1} \\ &= \frac{(\bar{\sigma}_x^{2,n-1})}{1+\bar{\sigma}_x^{2,n-1}/\sigma^2},\end{aligned} \tag{33}$$

where σ^2 is the variance of our measurement (we can make this depend on x, but here we are keeping the notation simple). Next we wish to find the change in the variance of our estimate of the mean. We define the change in the variance using

$$\tilde{\sigma}_x^{2,n} = \operatorname{Var}\left[\mu_x^{n+1} - \mu_x^n \mid S^n\right].$$

It is not hard to show that

$$\begin{aligned}\tilde{\sigma}_x^{2,n} &= \bar{\sigma}_x^{2,n} - \bar{\sigma}_x^{2,n+1} \\ &= \frac{(\bar{\sigma}_x^{2,n})}{1+\sigma^2/\bar{\sigma}_x^{2,n}}.\end{aligned} \tag{34}$$

It is useful to compare the updating equation for the variance (33) with the change in the variance in (34). The formulas have a surprising symmetry to them.

The knowledge gradient is found by first computing

$$\zeta_x^n = -\left|\frac{\mu_x^n - \max_{x'\neq x}\mu_{x'}^n}{\tilde{\sigma}_x^n}\right|.$$

ζ_x^n is the *normalized influence* of decision x. It is the number of standard deviations from the current estimate of the value of decision x, given by μ_x^n, and the best alternative other than decision x. We next compute

$$f(\zeta) = \zeta\Phi(\zeta) + \phi(\zeta),$$

where $\Phi(\zeta)$ and $\phi(\zeta)$ are, respectively, the cumulative standard normal distribution and the standard normal density. Finally, the knowledge gradient is given by

$$\nu_x^{KG,n} = \tilde{\sigma}_x^n f(\zeta_x^n).$$

TABLE 2. Calculations illustrating the knowledge gradient index.

Decision	μ	$\bar{\sigma}$	$\tilde{\sigma}$	ζ	$f(\zeta)$	KG index
1	1.0	2.5	1.336	−1.497	0.030	0.789
2	1.5	2.5	1.336	−1.122	0.066	1.754
3	2.0	2.5	1.336	−0.748	0.132	3.516
4	2.0	2.0	1.155	−0.866	0.107	2.467
5	3.0	1.0	0.707	−1.414	0.036	0.503

Table 2 illustrates the calculations for a problem with five choices. Choices 1, 2, and 3 have the same variance, but with increasing means. The table shows that the knowledge gradient index increases with the variance, demonstrating the behavior that we generally want to measure the best choice if the uncertainty is the same. Choices 3 and 4 have the same mean, but choice 4 has a lower variance. The knowledge gradient favors the choice with the highest level of uncertainty. Finally, choice 5 is the best, but because the level of uncertainty is the smallest, it receives the smallest index.

Frazier et al. [13] show that the knowledge gradient policy is asymptotically optimal (that is, in the limit as $n \to \infty$, it will find the best choice), and it is always optimal if there are only two choices. Furthermore, it is possible to bound the degree to which it is suboptimal, although these bounds tend to be weak. More significantly, the method has been shown experimentally to outperform other competing heuristics such as interval estimation.

9.3. Knowledge Gradient for Correlated Measurements

There are many problems where making one measurement tells us something about what we might observe from other measurements. For example, measuring the concentration of a chemical in a river at one location is likely to be correlated with measurements taking from other locations (particularly downstream). Observing the demand for an item at one price provides an idea of the demand for the item at other prices. Measuring the prevalence of a disease in one part of town hints at the prevalence in other areas of the same town.

Correlations are particularly important when the number of possible measurements is extremely large. The measurement might be continuous (where in the United States should we test birds for bird flu?), or there may simply be a very large number of choices (such as websites relevant to a particular issue). The number of choices to measure may be far larger than our budget to measure them.

The knowledge gradient concept is extended in Frazier et al. [12] to problems with correlated measurements. We will assume that we have a covariance matrix (or function) that tells us how measurements of x and x' are correlated. If x is a scalar, we might assume that the covariance of Y_x and $Y_{x'}$ is given by

$$\mathrm{Cov}(x, x') \propto \exp -\rho|x - x'|.$$

Or, we just assume that there is a known covariance matrix Σ with element $\sigma_{xx'}$.

To handle correlated measurements, we have to make the transition to working with vectors of means and covariance matrices. Let μ be the vector of means with element μ_x, and let Σ be the covariance matrix, with element $\mathrm{Cov}(x, x')$. There are three ways of writing the updated mean μ^n_x from measuring x^n. The first is the vector version of the Bayesian updating formulas, given by

$$\begin{aligned}\mu^{n+1} &= \Sigma^{n+1}\big((\Sigma^n)^{-1}\mu^n + (\lambda_{x^n})^{-1}W^{n+1}e_{x^n}\big),\\ \Sigma^{n+1} &= \big((\Sigma^n)^{-1} + (\lambda_{x^n})^{-1}e_{x^n}(e_{x^n})'\big)^{-1},\end{aligned}$$

where e_x is a column M-vector of 0s with a single 1 at index x, and $'$ indicates matrix transposition. The second way of writing the updating of uses a method for recursively updating the inverse of a matrix using something known as the Sherman-Woodbury equations. Using $x = x^n$, these are given by

$$\mu^{n+1} = \mu^n + \frac{W^{n+1} - \mu^n_x}{\lambda_x + \Sigma^n_{xx}} \Sigma^n e_x, \tag{35}$$

$$\Sigma^{n+1} = \Sigma^n - \frac{\Sigma^n e_x e_x' \Sigma^n}{\lambda_x + \Sigma^n_{xx}}. \tag{36}$$

The third method gives us a more convenient analytical form. Define the vector $\tilde{\sigma}$ as

$$\tilde{\sigma}(\Sigma, x) := \frac{\Sigma e_x}{\sqrt{\lambda_x + \Sigma_{xx}}}. \tag{37}$$

Also, let $\tilde{\sigma}_i(\Sigma, x)$ be the component $e_i'\tilde{\sigma}(\Sigma, x)$ of the vector $\tilde{\sigma}(\Sigma, x)$.

We note that $\mathrm{Var}^n[W^{n+1} - \mu^n] = \mathrm{Var}^n[\theta_{x^n} + \varepsilon^{n+1}] = \lambda_{x^n} + \Sigma^n_{x^n x^n}$, where $\mathrm{Var}^n[\cdot]$ represents the variance given what we know at iteration n, which is to say all the measurements up through W^n. Next define the random variable $Z^{n+1} := (W^{n+1} - \mu^n)/\sqrt{\mathrm{Var}^n[W^{n+1} - \mu^n]}$, where $\mathrm{Var}^n[W^{n+1} - \mu^n] = \lambda_x + \Sigma^n_{xx}$. We can now rewrite (35) as

$$\mu^{n+1} = \mu^n + \tilde{\sigma}(\Sigma^n, x^n) Z^{n+1}. \tag{38}$$

Note that after n measurements, $\mathbb{E}W^{n+1} = \mu^n$, which means that $\mathbb{E}Z^{n+1} = 0$. Also, it is easy to see that $\mathrm{Var}Z^{n+1} = 1$ (we constructed it that way). Because W^{n+1} is normally distributed, Z^{n+1} is normally distributed with mean 0 and variance 1. This will prove to be a more convenient form, as we see shortly.

The knowledge gradient policy for correlated measurements is computed using

$$\begin{aligned} X^{KG}(s) &\in \arg\max_x \mathrm{E}\Big[\max_i \mu_i^{n+1} \,\Big|\, S^n = s, x^n = x\Big] \\ &= \arg\max_x \mathrm{E}\Big[\max_i \mu_i^n + \tilde{\sigma}_i(\Sigma^n, x^n) Z^{n+1} \,\Big|\, S^n, x^n = x\Big], \end{aligned} \tag{39}$$

where Z is a one-dimensional standard normal random variable. The problem with this expression is that the expectation is hard to compute. We encountered the same expectation when measurements are independent, but in this case we just have to do an easy computation involving the normal distribution. When the measurements are correlated, the calculation becomes a lot more difficult.

One strategy is to approximate the expectation using Monte Carlo methods. This means that for each possible decision x, we perform repeated observations of Z. Let $Z(\omega)$ be a sample realization of the standard normal deviate. If we were programming this in Excel, we could do this using $Z = NORMINV(RAND(\,), 0, 1)$. We would do this N times and take an average. Of course, some experimentation would be needed to choose a good value of N. This simulation has to be done for each value of x. After this, we would choose the best value of x.

There is a way to compute the expectation exactly. We start by defining

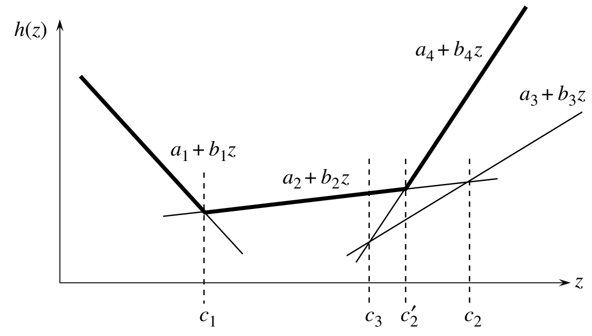
$$h(\mu^n, \tilde{\sigma}(\Sigma^n, x)) = \mathrm{E}\Big[\max_i \mu_i^n + \tilde{\sigma}_i(\Sigma^n, x^n) Z^{n+1} \,\Big|\, S^n, x^n = x\Big]. \tag{40}$$

Substituting (40) into (39) gives us

$$X^{KG}(s) = \arg\max_x h(\mu^n, \tilde{\sigma}(\Sigma^n, x)). \tag{41}$$

Now let $h(a, b) = \mathrm{E}[\max_i(a_i + b_i Z)]$, where $a = \mu_i^n$, $b = \tilde{\sigma}_i(\Sigma^n, x^n)$ and Z is our standard normal deviate. Both a and b are M-dimensional vectors. We next sort the elements b_i so

FIGURE 5. Regions of z over which different choices dominate. Choice 3 is always dominated.



that $b_1 \le b_2 \le \cdots$ so that we get a sequence of lines with increasing slopes. If we consider the lines $a_i + b_i z$ and $a_{i+1} + b_{i+1} z$, we find they intersect at

$$z = c_i = \frac{a_i - a_{i+1}}{b_{i+1} - b_i}.$$

For the moment, we are going to assume that $b_{i+1} > b_i$. If $c_{i-1} < c_i < c_{i+1}$, then we can generally find a range for z over which a particular choice dominates, as depicted in Figure 5. It is possible (as we see in the figure) that a choice is always dominated, which can be identified when $c_{i+1} < c_i$, as occurs in the figure for choice 3. When this happens, we simply drop it from the set. Thus, instead of using c_2, we would use c_2' to capture the intersection between the lines for choices 2 and 4. Once we have the sequence c_i in hand, we can compute (39) using

$$h(a,b) = \sum_{i=1}^{M} a_i(\Phi(c_i) - \Phi(c_{i-1})) + b_i(\varphi(c_{i-1}) - \varphi(c_i)).$$

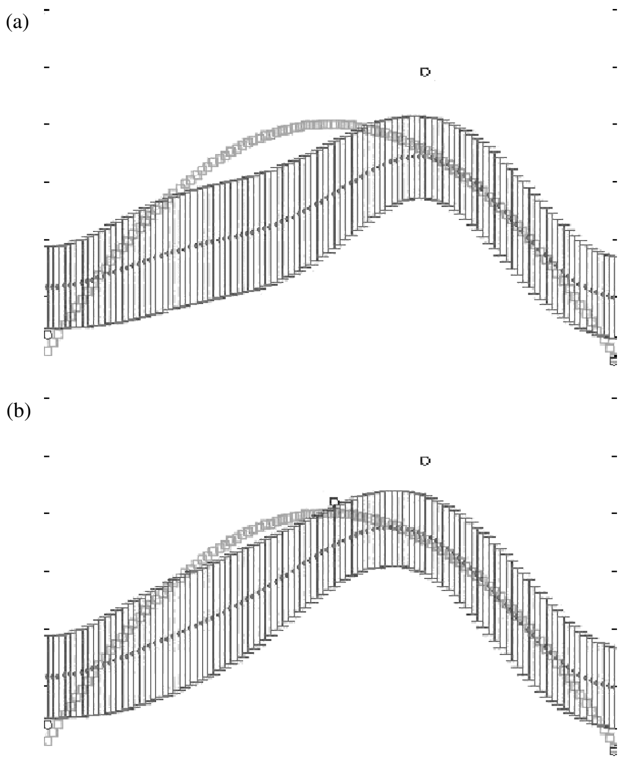
Of course, the summation has to be adjusted to skip any choices i that were found to be dominated.

Figure 6 illustrates the use of the correlated knowledge gradient algorithm when it is used to try to estimate a nonlinear function, starting with a constant confidence interval over the entire range. The logic begins by sampling the two endpoints. The third measurement is shown in Figure 6a, which occurs roughly at the 2/3 point (even though our best estimate of the maximum is at the midpoint). Figure 6b shows the fourth measurement, which is to the left of the third measurement (which corresponds roughly to where the current estimate of the maximum lies). The measurements illustrate that we are not choosing points that correspond to the highest point on the curve, but instead we are choosing the points where we have the best chance of improving our estimate of the maximum of the function.

The covariance matrix can be used to update the estimates of the means and variances (as we do in Equations (35)–(36)), and in the decision function (as is done in Equation (41)). This suggests three ways of collecting measurements and updating estimates: using the covariances in both the transition function (Equations (35)–(36)) and the decision function (Equation (41)), using the covariances in just the transition function (but assuming independence when we make a decision), and assuming independence in both the decision function and the transition function.

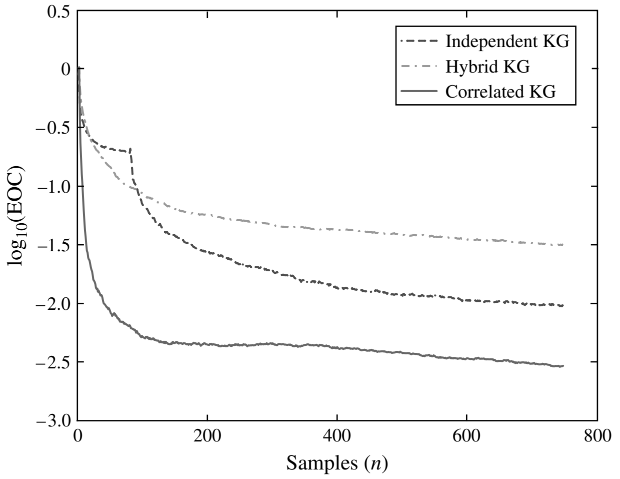
Figure 7 shows the log of the expected opportunity cost as a function of the number of measurements for the three policies described above. The "correlated KG" policy, which uses the covariance in both the decision function and the transition function, significantly outperforms the other two policies. Somewhat surprisingly, the second policy, which uses the covariance matrix in the transition function but not the decision function, performed the worst.

FIGURE 6. Estimate of a nonlinear function after (a) three measurements and (b) four measurements.



Handling correlations between measurements has tremendous practical value. When we assumed independent measurements, it was necessary to measure each option at least once. There are applications where the number of potential measurements is far greater than the number of measurements that we can actually make. If we have information about correlations, we can handle (in principle) a much larger number of measurements (even potentially infinite) by using correlations to fill in the gaps.

FIGURE 7. Expected opportunity cost as a function of the number of measurements, for each of three policies (from Frazier et al. [12]).



9.4. Knowledge Gradient for Online Learning

The knowledge is a particularly simple and elegant strategy for collecting information. It is near optimal and, of particular value, it can be adapted to handle correlated measurements. A natural question, then, is whether it can be adapted for online applications. In this setting, it is necessary to strike a balance between the reward we receive now, and the value of what we learn now on the future. Imagine that we have a finite horizon problem where we are going to make N decisions.

Let μ_x^n be our estimate of the value of choosing x given what we know after n measurements, and let ν_x^n be the knowledge gradient from a single additional measurement of x. Let K_x^n be the expected value of measuring x from $n+1$ up to N.

$$K_x^n = \mu_x^n + \mathbb{E}^n \sum_{m=n+1}^{N} [V^{m+1}(S^{m+1}(x)) - V^m(S^m) \mid S^n]. \tag{42}$$

We do not know how to compute K_x^n, but we can approximate it by

$$K_x^n \approx \mu_x^n + (N-n)\nu_x^n.$$

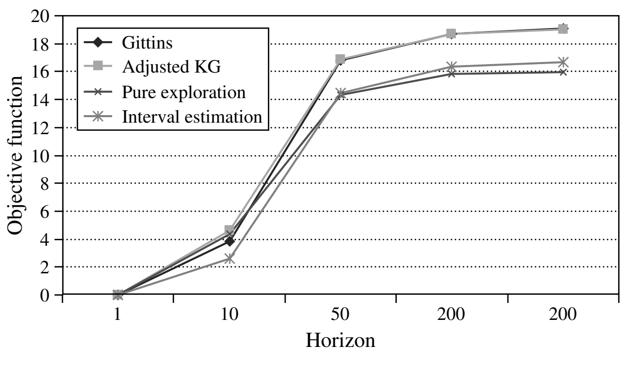
We suspect that this approximation is probably an upper bound on K_x^n. It strikes a balance between the reward now, μ_x^n, and the value of future rewards from what we learn now, approximated by $(N-n)\nu_x^n$. Thus, as we get close to the end of the horizon, we are less willing to measure a choice x with a low current reward in order to measure something later.

Figure 8 compares the use of Gittins indices to our online adaptation of the knowledge gradient for finite horizon problems. These were also compared against a pure exploration policy and a policy based on interval estimation. The experiments were run for horizons of 1, 10, 50, 100, and 200. The online adaptation of the knowledge gradient outperformed Gittins for small horizons; for $N = 10$, the knowledge gradient returned 4.62 versus 3.86 using Gittins indices. For larger number of iterations, the two procedures are very close (Gittins outperforms knowledge gradient for horizons N of 100 or more).

10. Summary

Learning arises in a broad range of application areas. This tutorial describes some of the dimensions and critical concepts that arise in learning problems, but the presentation is hardly comprehensive. There are classical information acquisition problems that are typically presented in the decision-tree literature that are not covered. The machine learning community has an active subfield known as active learning, which addresses these questions. There is, of course, the field of experimental design (which is primarily nonsequential), which we have not covered.

Figure 8. Comparison of adjusted KG for online measurements to Gittins indices, interval estimation, and pure exploration.



Despite the considerable attention that this problem has received, there is much that is not known or understood. Part of the problem is that these problems are often easy to formulate but impossible to solve. As a result, research has generally focused on heuristic policies with nice properties.

References

[1] R. E. Bechhofer, J. Kiefer, and M. Sobel. *Sequential Identification and Ranking Procedures.* University of Chicago Press, Chicago, 1968.

[2] R. E. Bechhofer, T. J. Santner, and D. M. Goldsman. *Design and Analysis of Experiments for Statistical Selection, Screening and Multiple Comparisons.* John Wiley & Sons, New York, 1995.

[3] M. Brezzi and T. L. Lai. Optimal learning and experimentation in bandit problems. *Journal of Economic Dynamics and Control* 27(1):87–108, 2002.

[4] A. Cayley. Mathematical questions with their solutions, no. 4528. *Educational Times* 23(18), 1875.

[5] C. H. Chen, L. Dai, and H. C. Chen. A gradient approach for smartly allocating computing budget for discrete event simulation. *Simulation Conference Proceedings, 1996. Winter* 398–405, 1996.

[6] H. C. Chen, C. H. Chen, and E. Yucesan. Computing efforts allocation for ordinal optimization and discrete event simulation. *IEEE Transactions on Automatic Control* 45(5):960–964, 2000.

[7] C. H. Chen, J. Lin, E. Yücesan, and S. E. Chick. Simulation budget allocation for further enhancing the efficiency of ordinal optimization. *Discrete Event Dynamic Systems* 10(3):251–270, 2000.

[8] H. C. Chen, L. Dai, C. H. Chen, and E. Yücesan. New development of optimal computing budget allocation for discrete event simulation. *Proceedings of the 29th Conference on Winter Simulation*, 334–341, 1997.

[9] S. E. Chick and K. Inoue. New procedures to select the best simulated system using common random numbers. *Management Science* 47(8):1133–1149, 2001.

[10] S. E. Chick and K. Inoue. New two-stage and sequential procedures for selecting the best simulated system. *Operations Research* 49(5):732–743, 2001.

[11] S. E. Chick, J. Branke, and C. Schmidt. New myopic sequential sampling procedures. Submitted to *INFORMS Journal on Computing*, 2007.

[12] P. Frazier, W. B. Powell, and S. Dayanik. The knowledge gradient policy for correlated normal rewards. Working paper, Princeton University, Princeton, NJ, 2007.

[13] P. Frazier, W. B. Powell, and S. Dayanik. A knowledge gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, forthcoming, 2008.

[14] M. C. Fu, J.-Q. Hu, C.-H. Chen, and X. Xiong. Simulation allocation for determining the best design in the presence of correlated sampling. *INFORMS Journal on Computing* 19(1):101–111, 2007.

[15] J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society, Series B*, 14:148–177, 1979.

[16] J. C. Gittins. *Multi-Armed Bandit Allocation Indices.* John Wiley & Sons, New York, 1989.

[17] J. C. Gittins and D. M. Jones. A dynamic allocation index for the sequential design of experiments. J. Gani, ed. *Progress in Statistics* 241–266, 1974.

[18] S. S. Gupta and K. J. Miescke. Bayesian look ahead one-stage sampling allocations for selection of the best population. *Journal of Statistical Planning and Inference* 54(2):229–244, 1996.

[19] M. Hartmann. An improvement on Paulson's procedure for selecting the population with the largest mean from k normal populations with a common unknown variance. *Sequential Analysis* 10(1–2):1–16, 1991.

[20] D. H. He, S. E. Chick, and C. H. Chen. Opportunity cost and OCBA selection procedures in ordinal optimization for a fixed number of alternative systems. *IEEE Transactions on Systems, Man and Cybernetics Part C-Applications and Reviews* 37(5, September):951–961, 2007.

[21] L. P. Kaelbling. *Learning in Embedded Systems.* MIT Press, Cambridge, MA, 1993.

[22] S.-H. Kim and B. L. Nelson. A fully sequential procedure for indifference-zone selection in simulation. *ACM Transactions on Modeling and Computer Simulation*, 11(3):251–273, 2001.

[23] B. Nelson, J. Swann, D. Goldsman, and W. Song. Simple procedures for selecting the best simulated system when the number of alternatives is large. *Operations Research* 49(6):950–963, 2001.

[24] E. Paulson. A sequential procedure for selecting the population with the largest mean from k normal populations. *The Annals of Mathematical Statistics*, 35(1, March):174–180, 1964.

[25] E. Paulson. Sequential procedures for selecting the best one of k koopman-darmois populations. *Sequential Analysis* 13(3), 1994.

[26] W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality.* John Wiley & Sons, New York, 2007.

[27] M. L. Puterman. *Markov Decision Processes.* John Wiley & Sons, New York, 1994.

[28] H. Raiffa and R. Schlaifer. *Applied Statistical Decision Theory.* MIT Press, Cambridge, MA, 1968.

[29] Y. Rinott. On two-stage selection procedures and related probability-inequalities. *Communications in Statistics—Theory and Methods* 7(8):799–811, 1978.

[30] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvari. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning* 38(3):287–308, 2000.

[31] Y. C. Yao. Some results on the Gittins index for a normal reward process. *IMS Lecture Notes—Time Series and Related Topics* 52:284–294, 2006.