Risk Analysis, Vol. 37, No. 6, 2017 DOI: 10.1111/risa.12834

Book Review

Algorithms to Live By: The Computer Science of Human Decisions, Brian Christian and Tom Griffith. 2016. Picador: New York, NY
Reviewed by Tony Cox

Risk analysis provides methods for using data to predict the probabilities of various consequences for different choices or policies under uncertainty. Risk management typically encourages the selection of options with high expected utility and low expected regret, when these can be identified, although it is now well understood that real people deviate systematically and predictably from such recommendations in many risk management settings. Might machines do better? Could artificial intelligence and operations research optimization algorithms confronted with realistically noisy and imperfect data and incomplete understanding of how actions affect outcomes nonetheless produce high-quality decision recommendations? If so, what performance guarantees could be attached to them, and how complex would the required algorithms be? What lessons might human decisionmakers learn from the most successful of these algorithms that might be applied to daily life? These questions are addressed in the highly readable popular book *Algorithms to Live By: The Computer Science of Human Decisions*, by Brian Christian and Tom Griffiths (Henry Holt and Company, New York, 2016). The answers shed valuable, and often surprising, light on how uncertainty can simplify decision making and make otherwise computationally intractable decision problems easier to solve. They show how human judgment, priority setting, forecasting, and behaviors can be improved—meaning spending less time and effort to achieve better results on average—by adopting principles used in computer science to improve the performance of computers and networks.

The book consists of an introduction, 11 chapters, and a conclusion. The introduction explains that "there is a particular set of problems that all people face, problems that are a direct result of the fact that our lives are carried out in finite space and time. What should we do, and leave undone, in a day or in a decade? What degree of mess should we embrace—and how much order is excessive? What balance between *new* experiences and *favored* ones makes for the most fulfilling life? These might seem like problems unique to humans; they're not. For more than half a century, computer scientists have been grappling with, and in many cases solving, the equivalents of these everyday dilemmas." The rest of the book explores how lessons from operations research and computer science can be used to improve responses to such challenges in everyday life, recognizing that "tackling real-world tasks requires being comfortable with chance, trading off time with accuracy, and using approximations."

A sense of its contents can be gleaned by sampling some of the main insights from each chapter, as follows.

Chapter 1: Optimal Stopping. Suppose that one is searching for a rare prize—perhaps a good parking spot in a long line of cars, a best offer or candidate or opportunity in a sequence with qualitative or quantitative assessment of their values, or a drug lead to select for further development. When should one stop searching and commit to the current opportunity? In many cases, simple decision rules exist for optimizing when to accept a current candidate, opportunity, or offer vs. rejecting it in hopes of finding a better one. One famous rule for maximizing the probability of selecting the best of a known number of candidates is to examine the first 37% of the candidate pool and then accept the next candidate that is better than any of them. If quantitative evaluations are possible and the distribution of values is known, then the expected value of the selected opportunity can be maximized by accepting the first one greater than a threshold value, where the value of the threshold declines with the number of remaining candidates or opportunities. There is evidence that most people stop too soon in such problems and could increase their average rewards by searching somewhat longer.

Chapter 2: Explore/Exploit. When should a clinical trial of a new drug or treatment be suspended because enough evidence has accumulated to conclude that it is more effective (or less effective) than the incumbent? How much A/B testing should Internet companies do before committing to one version rather than another of a website or advertisement? How often should we revisit favorite restaurants that usually provide good experiences versus trying new ones that might prove substantially better or substantially worse? When should animals stop

 0272-4332/17/0100-1201$22.00/1

foraging in one spot and try another? More generally, how should one allocate time or effort between exploring new activities or opportunities with uncertain rewards and exploiting ones that, to date, appear to have the best reward distributions? Chapter 2 examines the solutions to such "multiarm bandit" decision problems and summarizes recent advances. It presents the important principle of *optimism in the face of uncertainty*, which prescribes always choosing next the activity with the highest upper confidence bound (UCB) for its uncertain rewards, given all the results of all the trials of different choices made so far. This UCB algorithm delivers approximately optimal decisions in the sense that the difference between the expected cumulative rewards that could have been achieved with perfect information about reward distributions and the expected cumulative rewards actually achieved (called the regret) grows only logarithmically, that is, about as much in the first 10 tries as in the next 90, or as in the next 900 after that. No decision algorithm can achieve lower growth in regret. Without the guidance of such algorithms, most people tend to overexplore in experimental settings, sampling different options for too long, rejecting inferior choices and homing in on the best ones too slowly, and therefore accumulating unnecessary regret. The authors suggest that exploration in youth and exploitation in old age may be a practical implication of UCB-type algorithms for helping to lead a minimum-regret life.

Chapter 3: Sorting. Sometimes, finding the best of a set of options by some criterion is not enough: one also wants to know the second-best, the third-best, and the entire rank-order of the different choices. Practical tasks and technology that depend on efficient sorting algorithms range from Google searches that present the few most relevant (top-ranked) webpages in response to a user query to efficient routing and distribution of mail or packages to reshelving library books returned at random times. More intriguingly, establishing dominance hierarchies in human or animal groups and designing tournament ladders or other competition structures in sports can be viewed in terms of efficient sorting based on a minimal number of comparisons. Chapter 3 surveys sorting algorithms that are well known to computer scientists, from the obvious but inefficient bubble sort based on repeated pairwise exchanges to merge sort, which actually achieves the theoretically optimal scaling behavior of taking computer time proportional to $n\log(n)$ to sort n items—far faster than bubble sort for large lists. The authors point out a fundamental tradeoff between time spent sorting now and time spent searching or retrieving later and suggest that people and organizations often tolerate less messiness than is optimal, spending precious time organizing e-mail files or other items that will rarely be searched and for which searching would in any case take less time than sorting.

Chapter 4: Caching. Computers, organizations, and individuals must manage the risks of moving items from more-accessible locations to less-accessible locations (such as a to a relatively slow hard drive instead of much faster RAM) when there are too many to items for them all to be kept in the most accessible location and when the time at which an item will next be needed is uncertain. Various plausible-sounding strategies have been proposed for deciding what to keep at hand and what to move to longer-term storage, such as first-in, first-out (FIFO), which evicts the oldest item first when something must be cleared from the most-accessible locations to make room for new items. A simple heuristic that is hard to beat in general for this type of decision making under uncertainty is the least-recently used (LRU) heuristic. This evicts from readily accessible locations whichever item has gone the longest without being retrieved. How long it has been since we last needed to use a resource is often the most practical predictor of how long it will be until we will need it again, at least in contexts where the more recently an item has been used, the more likely it is to be used soon again. This principle applies in many contexts outside of computer science. Amazon uses the LRU principle to proactively ship items that have recently been popular in a region to a warehouse in that region, anticipating that items that have been demanded recently are likely to be demanded again soon. This is part of how data on recent usage (or recent deliveries, for Amazon) can be used to adaptively position scarce resources at locations where they are most likely to be needed, thus reducing service or delivery times and minimizing risks of slow or late deliveries.

Chapter 5: Scheduling. Many risk analysts work under conditions where there seems always to be more work to be done than resources available to do it quickly. A long list of Superfund sites to be addressed or of chemicals to be scrutinized under REACH or similar regulatory programs provides a constant backlog of problems that can keep available resources fully occupied addressing them. Under such conditions, how should the limited person-hours available be allocated most effectively to maximize the value of the resulting flow of benefits from sites remediated, chemicals investigated, or other tasks completed? The answer depends on exactly what one's goals are. To minimize the maximum lateness of any task on a list with desired deadlines, one should always work next on the task with the earliest due date. But to minimize the average completion time per task, one should instead work next on the task with the shortest time to completion—the shortest processing time (SPT) rule. If there is time pressure because some sites are deteriorating, and if the goal is to minimize the number of sites that deteriorate past some threshold level, then a procedure called Moore's Algorithm should be followed; this identifies and skips the sites that would consume so many resources that many other sites would then deteriorate unacceptably. For many purposes, allocating available effort to maximize a simple criterion, expected benefit received per unit of time spent (or "bang for the buck," in more colloquial terms), will make the most productive use of scare resources. There is evidence that animals allocate their foraging or hunting time according to this rule.

Interestingly, slight changes in the exact goal that one is trying to achieve, such as allocating resources to minimize the number of sites that deteriorate to an unacceptable level versus allocating them to minimize the sum of quantitative damages done by such sites, can turn the allocation from one that is easy to solve to one for which there is no efficient solution algorithm. Most known resource-scheduling problems (about 90%) cannot be solved by any computationally tractable algorithms if there is perfect information about what needs to be done by when. The various principles that work well for simple problems, such as the SPT rule, then become the basis for heuristics that may be the best that can be done in practice. Remarkably, these simple rules may become optimal, or close to it, when there is uncertainty about what will need to be done by when due to randomly arriving additional tasks such as newly identified sites to inspect or chemicals to be screened. Uncertainty turns simple prioritization rules, especially the principle of always working next on the available task with the greatest ratio of expected benefit achieved per person-hour spent, into formulas for adaptively allocating time as new tasks or opportunities arise at random. They can be optimal, or nearly so, for deciding how to allocate time to tasks under uncertainty to maximize the value of the flow of benefits produced, even if no computationally tractable procedure exists that would be optimal or nearly optimal if all tasks and deadlines were known in advance.

Chapter 6: Bayes's Rule. Suppose that we want to predict some uncertain quantity—how much longer a drought will last, how much longer a sick friend will live, how much money a new movie will make, or how many more terms a politician will remain in office—based on the evidence provided by the value that the uncertain quantity has reached so far—how long the drought has already lasted, how much money the movie has already grossed, how long the friend has already lived since diagnosis, how long the politician has already been in office. Bayesian inference treats all uncertain quantities as random variables and conditions the original ("prior") distribution for any uncertain quantity on the available evidence to obtain an updated ("posterior") distribution for predicting its value. Quantitatively, as the authors explain, Bayes's Rule "gives a remarkably straightforward solution to the problem of how to combine preexisting beliefs with observed evidence: multiply their probabilities together." This often leads to simple, useful rules for estimating and predicting uncertain quantities and for updating the results as new data become available. For example, if a bus has been late on x of the past n occasions, then the probability that it will be late next time can be estimated as $(x + 1)/(n + 2)$ ("Laplace's Law"). Because they write for a general audience, Christian and Griffiths do not detail all of the assumptions behind such simple rules (e.g., uniform prior, binomial sampling, stationary distribution, use of the mean of the beta posterior as a predictor), but they motivate and explain the key results and their supporting ideas succinctly and well. For forecasting, they note three main types of underlying prior distributions for the time until an event occurs: power law, normal, or Erlang. These imply three different types of forecasting rules. Power-law ("scale-free") distributions lead to multiplicative forecasts. For example, the longer one has been waiting in a call queue to speak to a customer service representative, the longer the remaining wait is expected to be: expected remaining time increases in proportion to the time already spent. Normal (or other single-peaked) prior distributions for waiting time imply that the average time is a good basis for prediction and expected remaining wait decreases with time already spent. Exponential waiting times imply that the expected remaining wait is constant no matter how much time has already elapsed, perhaps justifying repeated claims (spaced more than 5 minutes apart) that "I just need 5 minutes more!" from someone trying to finish a task.

Drawing in part on their own research, the authors note that people are often excellent intuitive forecasters, giving estimates for remaining waiting times very close to those from Bayes's Rule, but only if they understand the prior distribution accurately: "Good predictions require good priors." With good priors, even a small amount of data, as little as a single observation, can often be used to obtain accurate forecasts. Without them, our expectations and experiences will often conflict. The authors note that exposure to news media reports can greatly distort beliefs about risks and their base rates and that being a good intuitive Bayesian—a person whose expectations generally match real-world outcomes—may require protecting our experience-based priors from such distortions.

Chapter 7: Overfitting. When can decision making under uncertainty be improved by ignoring available information? Bayesian decision theory suggests that conditioning on more information can never reduce the quality of a decision—in principle, information never has negative value—but both psychology and machine learning teach a different lesson. People and statistical algorithms are prone to pay too much attention to irrelevant details that distract from the key factors that drive outcomes. This leads to "overfitting," i.e., construction of overly elaborate models that describe details of past data better than simpler ones but that have less predictive power because they do not generalize well. The advent of big data feeds this tendency as researchers look for patterns in data without always understanding the underlying data-generating processes well enough to know which patterns can be generalized to future situations. Fortunately, machine learning algorithms can be made to avoid overfitting by penalizing model complexity. In addition, hold-out samples can be used to assess and improve the out-of-sample performance of algorithms trained on a subset of the data. The authors suggest that such cross-validation might also be useful is human schools and organizations to reduce training-to-the-test or optimizing only what is measured rather than what is intended.

Psychologically, the literature on "fast, frugal heuristics" shows that simple rules of thumb often outperform more complicated approaches to decision making in the real world. For example, to predict the probability that someone played basketball in college, one might start by asking whether height is more or less that six feet; the answer to that single question already provides useful information for discriminating between people who are more likely and less likely to be basketball players. Conditioning on the answers to a handful of such simple questions often provides predictions that are about as accurate as possible based on the data and that are more robust and accurate than those from more complex statistical models that include many more factors. At the intersection of human and machine decision making is an Occam's razor-like principle: focusing on the few most important factors that drive outcomes and using simple, robust if-then rules to interpret them can simplify the construction *and* improve the performance of data-driven prediction and decision rules.

Chapter 8: Relaxation. The theme that simpler can be better for decision making can also be applied in a different way: rather than solving a very difficult combinatorial optimization problem to decide how best to allocate scare resources to accomplish a task in the presence of multiple constraints, it may be possible to solve a simplified version of the problem in which one or more constraints is removed (or "relaxed") and then tweak the solution a little, for example, by rounding fractional answers to the nearest whole number to decide how many discrete units of a resource to allocate. Solving modified problems with relaxed constraints often yields an approximate solution to the original problem. Some relaxation techniques even provide performance guarantees that the solution obtained from the simplified problem will not be worse (e.g., more costly or time consuming) than the true but hard-to-find optimal solution to the original problem by more than a known amount. Confronted with a difficult or computationally intractable decision problem such as how to allocate spare parts or other costly resources to optimize the performance (e.g., net benefit per unit time) of a complex reliability system, a decisionmaker may be able to use relaxation to decide when a good enough solution has been found. Since the optimized value for the original problem with all constraints enforced can never be better than the optimized value for a relaxed problem with some constraints removed, one can stop searching for a better solution to the original problem if the value for the best solution discovered so far is close to the value for the exact solution to the relaxed problem. Further improvement efforts cannot produce further benefits larger than the gap between the value of the current best known solution and the value of the exact solution to the relaxed problem.

Chapter 9: Randomness. A different strategy for solving computationally difficult or impossible decision optimization problems to any desired degree of precision is to use randomized algorithms. For example, Monte Carlo simulation makes short work of estimating expected values with as much precision as desired, even if the distribution being sampled from is complex. It can be applied to model the probabilistic behavior and performance of a complex system operating in a complicated and uncertain environment under different risk management policies, provided that the relevant uncertainties are understood well enough to be simulated by sampling from known distributions and then computing functions of the sampled values. Randomized algorithms such as simulated annealing, even more than relaxation, allow practical nearly-optimal solutions to be discovered to challenging discrete optimization problems such as resource allocation decision problems in which the number of possible decision alternatives is far too large to allow each one to be evaluated. As the authors explain: "A close examination of random samples can be one of the most effective means of making sense of something too complex to be comprehended directly. When it comes to handling a qualitatively unmanageable problem, something so thorny and complicated that it can't be digested whole—solitaire or atomic fission, primality testing or public policy—sampling offers one of the simplest, and also the best, ways of cutting through the difficulties."

Chapter 10: Networking. Chapter 10 asks how people and computers can coordinate their messages and acknowledgments and adjust their expectations and use of shared resources to enable reliable communications over networks of failure-prone and unreliable components. It explains two main technical principles, exponential backoff and additive increase, multiplicative decrease (AIMD), originally developed for routing packets reliably through telecommunications networks even when no sender knows how much network capacity others will need, or when. In this distributed control context, exponential backoff refers to each sender doubling the expected time to retry a transmission whenever a conflict is encountered. Actual retransmit times are uniformly distributed between one period and a maximum time that is doubled after each failure because randomization helps to make conflicts less likely. AIMD refers to a policy of ramping up a sender's transmission rates of packets quickly, doubling the rate at each time step and cutting back by half as soon as a conflict occurs because a conflict signals that another user is seeking to share the available capacity; thereafter, the transmission rate is increased by only one extra packet per period ("additive increase") as long as no conflict occurs, but is halved ("multiplicative decrease") whenever a conflict occurs. A key reason that the Internet functions as well as it does without centralized control is that AIMD is built into the transmission control protocol for packet-switched networks. But Christian and Griffiths see many opportunities to apply these principles beyond the confines of telecommunications networks. They note that in human relationships as well as in societal punishment of criminal behavior, we tend to give unreliable persons—ones who do not reliably reply to invitations or who repeatedly violate the terms of probation—a certain number of chances and then give up on them, perhaps dropping an unreliable friend or imprisoning a probationer: "Three strikes and you're out!" Exponential backoff suggests an alternative, described by the authors as "finite patience and infinite mercy," in which the time between successive invitations or the amount of jail time for each probation violation starts small and is successively (and predictably) increased if no answer is received or if another probation violation occurs. In practice, a five-year study of such a program in Hawaii found that probationers treated with this protocol were half as likely as regular probationers to be arrested for a new crime or to have their probations revoked; 17 states have subsequently adopted similar programs. Ant colonies and foraging animals use AIMD-like strategies in which success is met with ramp-up and failure with cut-back. As the authors state: "More broadly, AIMD suggests an approach to the many places in life where we struggle to allocate limited resources in uncertain and fluctuating conditions," helping to make the most efficient use of available resources while managing uncertainty about their availability and performance.

Chapter 11: Game Theory. The final chapter examines problems of coordination and conflict from the perspective of incentives, information, and computational complexity. It begins with the striking thesis that core concepts such as Nash equilibria that provide the intellectual foundation for much of modern economics, political economy, and game theory have little or no practical value for predicting behaviors when there is no effective way to compute them—as is often the case. Enumeration and inspection of all possibilities suffices to identify pure-strategy equilibria, if there are any, in simple games where each player has only a few possible actions. But no effective algorithms exist for finding Nash equilibria of general games, for determining which specific actions should be used in those equilibria, or even for determining whether there are multiple equilibria. Such computational limitations challenge the practical applicability of many theoretical results based on the assumption that all participants in a game will play Nash equilibrium strategies.

Christian and Griffiths see such computational limitations as challenging the very concept and definitions of rationality conventionally used in economics and in decision and risk analysis: a strategy or decision rule that maximizes expected utility cannot be considered the rational thing to do if there is no practical way to compute it with the time and resources available for decision making. In such cases, the fast frugal heuristics discussed in Chapter 7 may become the closest practicable approximation to rational behavior that real people can hope for. Chapter 11 also discusses the problem of "information cascades" in highly connected societies where the decisions of bidders in an auction, speculators in a stock market, or bettors in a prediction market swiftly flow into public information that may affect the beliefs and decisions of others, leading to collectively harmful bubbles and false expectations emerging from self-reinforcing feedback loops between public information and private decisions.

More constructively, Chapter 11 describes how game theory has been used to quantify the "price of anarchy" in systems where many participants interact. This is the ratio of (a) the value of a performance metric such as average cost or average waiting time per person when each participant individually decides what to do; to (b) the value of the same metric if a well-informed beneficent centralized controller coordinates their activities. For some important real-world applications such as traffic congestion during rush hour, the price of anarchy turns out to be only 4/3, implying that even if a centralized planner were to direct the movements of each vehicle (as might become possible in a future of self-driving cars), the average time for a commute could not be reduced by more than 25%. On the other hand, there are many situations, including those with Prisoner's Dilemma or Tragedy of the Commons incentives, where the price of anarchy is very high: the outcome when people act according to the incentives of the situation leads to bad outcomes for everyone. The potential for collective gains from centralized coordination and control by law, regulation, custom, religious authority, moral imperatives, executive edicts, or other mechanisms is correspondingly high in such situations, providing a strong rationale for government or other institutions and interventions.

Alternatively, game theory has been used to create algorithms for (re)designing incentive systems (called "mechanisms") to make the price of anarchy as small as possible. The authors suggest that perhaps emotions—from the helpless "irrational" rage that emboldens a small person or animal to attack a larger one that invades its rights or territory to the love that binds families or the solidarity that engenders acts of selfless altruism and courage in groups and communities—result from evolutionary mechanism design: changing the incentives felt in certain situations subordinates self-seeking to acts promoting group coordination, cohesion, and survival.

The book ends with a brief conclusion that encourages "computational kindness" in the design of systems and the conduct of daily affairs—that is, applying design principles, communications, and behaviors that make it easy for others to compute what is best to do in each situation. This concept is beautifully explained by the example of designing parking garages as spirals that take drivers further and further from the entrance. In contrast to other designs, this one makes the optimal search and stopping rule trivial to compute: drive until the first empty spot is encountered and then take it. Empirically, any extra time spent walking because of this design is more than compensated by savings in time spent searching.

Algorithms to Live By is a book of well-explained big ideas with important applications. Many of both the ideas and their applications are derived from highly technical literatures in machine learning, computer science, operations research, and telecommunications engineering. The authors aspire to make these ideas accessible to a wide public of nonspecialist readers, and in this they succeed admirably. The prose is lively and engaging throughout, every key concept is illustrated with real-world examples, and the constant interpretation of principles in terms of applications in everyday life makes them understandable and vividly reveals their practical importance. Along the way, the reader is treated to tales of the early days of the company that became IBM, whose card-sorting machine was predicted in 1890 to have very little profit potential; of physicians struggling to decide whether a new life-saving treatment really improved the survival of infants with respiratory failure (it did, but that finding took tragically long to become established convincingly in the age before adaptive trials based on multiarm bandit algorithms); and of a variety of other intriguing and instructive innovations and their inventors. These memorably demonstrate not only the importance and power of better ideas for securing better outcomes in an uncertain world, but also the challenges of introducing innovations into established systems that are not yet looking for better ways to accomplish their daily operations.

Algorithms to Live By would make fascinating and worthwhile supplemental reading for a variety of undergraduate and graduate courses in statistics, operations research, computer science, applied probability, decision analysis, and risk analysis. Although aimed at a general audience, it is backed by over 50 pages of notes and close to another 20 pages of bibliography, with many notes and references addressing the primary research literature. Its broad scope makes it stimulating even for more technical audiences. Moreover, the book makes a valuable contribution by challenging and expanding traditional concepts of rationality to take account of computational effort. In doing so, it adds precision and detail to earlier notions of bounded rationality by illustrating with concrete algorithmic examples how solution quality, computational time, and certainty must be traded off against each other in solving real-world decision problems.

This computational view of rational decision making under real-world constraints of uncertainty and complexity is expressed well in the book's conclusion, as follows: "The intuitive standard for rational decision-making is carefully considering all available options and picking the best one… [But] life is just too complicated for that. In almost every domain we've considered, we have seen how the more real-world factors we consider—whether it's having incomplete information when interviewing job applicants, dealing with a changing world when trying to resolve the explore/exploit dilemma, or having certain tasks depend on others when we're trying to get things done—the more likely we are to end up in a situation where finding the perfect solution takes unreasonably long. And indeed, people are almost always confronting what computer science regards as the hard cases. Up against such hard cases, effective algorithms make assumptions, show a bias toward simpler solutions, trade off the costs of error against the costs of delay, and take chances. These aren't the concessions we make when we can't be rational. They're what being rational means."