Review by Mark A. Kon, Boston University

Complexity and Information, by J.F. Traub and A.G. Werschulz, Cambridge University Press, Cambridge, 1998, 139 pp.

Continuous complexity theory gets its name from the model of mathematical computation on which it is based. In the mathematics of the standard combinatorial computational complexity theory, modeling the inner workings of a computer takes center stage, and complexities of problems are measured in terms of bit operations using the Turing machine model, leading to hierarchies of complexity and to problems such as the famous $P = NP$ conjecture. In the continuous complexity model, real number operations such as multiplication and function evaluation are primitives, and complexity analysis has a more analytic nature. The continuous (or real-number) model of computation is applicable for a number of reasons. One is the expectation that in current and future computations, many fundamental analytic operations (such as various types of function evaluations) can be carried out in short enough times and within close enough time orders of each other that other higher order computational factors (of which there can be many) dominate complexity analyses. The real-number model is also useful because it permits us to use the power of analysis in the study of complexity theory. This model of computation is the basis of recent foundational work in computation theory ([BSS], [BCSS]), and of older work ([T1], [HS], [S]) on zero finding for polynomial and other equations. Mathematical analysis takes the helm as the primary tool instead of the combinatorial techniques in classical computational complexity.

A second sometimes important element of the continuous theory is the notion of partial information in mathematical computation, which is the primary topic of this book. Computations in numerical analysis attempt to model infinite dimensional objects (e.g., a function f on a domain Ω), while computers (in the real number model) deal with finite dimensions (finite sets of reals). Thus to solve an elliptic PDE $Au = f$ with A a differential operator and f defined on Ω, the computer uses a finite set of real values $\{L_i(f)\}_{i=1}^n$ depending on f. The numbers $L_i(f)$ might range from values $f(x_i)$ at a fixed set $\{x_i\}_{i=1}^n$ of points (so-called *standard information*), to integral functionals $\int_0^1 f(x)e^{-2\pi ix}dx$ of f arising in a Fourier expansion. The computer processes the array $Nf = (L_1(f), \ldots, L_n(f))$ (information), and returns another finite set of reals, used to construct the unknown u. We remark that among the many methods of solving elliptic problems numerically, it has been proved in the context of continuous complexity that the so-called finite element method is optimal in a well-defined way (see [W]). One of the interesting aspects of continuous complexity theory is its use of functional analysis to study uncertainty and its foundations.

Computational problems can be reduced to computing an output from an input. Define S to be the mapping from input to desired output. For example input may consist of a function f on the unit interval. Information Nf about f might be values of f at a discrete set of points, a common information set in numerical analysis. Desired output

Sf might range from the integral Int f = $\int_0^1 f(x)\,dx$, to the recovery of f itself (from partial information Nf).

The notion of partial information is central in computational solutions of mathematical problems. Indeed there are only two ways to get one's hands on the functions and operators in numerical problems for which one seeks computational solutions. The first is to describe them analytically or symbolically, manipulate them in exact form until an answer is obtained, and finally to extract the finite collection of numbers we need to work with. The second and more common way is to start with information Nf about inputs (e.g., functions and operators) in partial form, as numbers we must do something with. Thus N has filtered f into the finite dimensional form Nf. How we now deal with this to get the final numerical approximation $\phi(Nf)$ of Sf is up to us. In neural network theory the so-called backpropagation and Boltzmann machine algorithms as well as others for programming networks are effectively founded on the above partial information model. In learning an unknown input-output (i-o) function f, a network obtains information about f using examples consisting of inputs x_i and their correct outputs $f(x_i)$. Using a learning algorithm, it then embodies the approximation of f which best extrapolates the examples to untested inputs x.

Effectively, the partial information paradigm can be seen to embody the question, How can we optimally filter an infinite dimensional map (say $S = A^{-1} : f \to u$) through finite dimensional spaces (to implement the map on a computer)? Equivalently if $S : F \to G$ is the map of interest (say F and G are normed linear spaces), we wish to find a finite dimensional Y and information operator N (along with a ϕ) such that the diagram

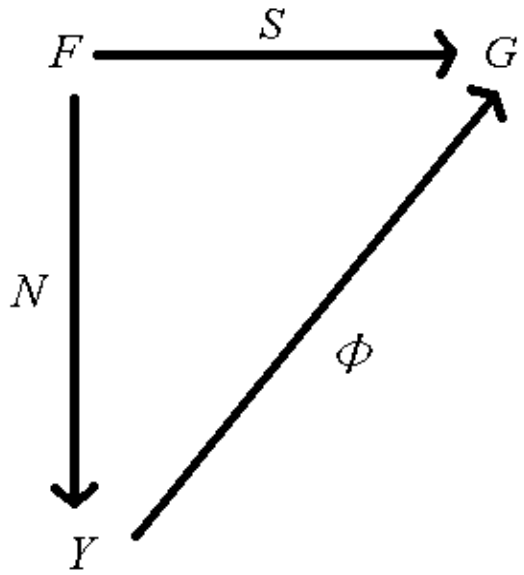

Fig. 1

commutes maximally. Above ϕ is the algorithm constructing an approximation $\phi(Nf) \approx Sf$ from the useable finite dimensional information Nf.

Measures of complexity in the real number model depend on the components N and ϕ of the approximation process. The complexity of N at its simplest can be defined as proportional to the rank n, the number of function evaluations. The complexity of the algorithm ϕ can also be gauged using appropriate criteria. In work on complexity of polynomial root finding and in the so-called BSS [BCSS] real model of computation, information is assumed full (effectively N is the identity), and the focus is on complexity of ϕ.

An interesting question in the context of continuous complexity is, what are the limits of computation for the approximation of a "difficult" output Sf from input f? Equivalently, what are lower bounds on the complexity of such an approximation (say within ϵ in an appropriate norm) using *any* computational method (i.e., any choice of N and ϕ) limited only by a fixed bound on complexity (tractability)? If we can make appropriate a priori assumptions about f, e.g., regarding smoothness, there are theorems which bound from below the complexity of any ϵ-approximation.

Obtaining the information Nf about f is often the rate limiting step in the production of the approximation of Sf. The analogy to thermodynamics made in the book is worth mentioning: in thermodynamics upper bounds are placed on energy output from systems involving pairs of heat baths at different temperatures, independent of what methods are used. Crackpots have tried in the past to "beat" the thermodynamics laws, so far with no success. Similar bounds exist in computation, with limitations also imposed by specific laws. If we do not have enough information (i.e., the rank n of Nf is too small), there is no way our worst case error $\|Sf - \phi(Nf)\|$ can be decreased below a fixed ϵ depending on n. These are rock-solid lower bounds on the amount of work for solving problems, which apply to all potential solution methods and so are difficult to obtain. Upper bounds are typically obtained by finding working algorithms. Lower bounds tell us where *not* to look for improvements on complexity: there is no benefit to looking for better information (e.g. a better choice of n points at which to sample a function) if we are already near a lower complexity bound.

An important example of the tractability issues the book addresses is the "curse of dimensionality". This can be blamed for problems ranging from the difficulty of building a neural network which can compute desired outputs from inputs, to the difficulty of predicting fluctuations of stock markets in mathematical finance. A typical problem occurs in the latter area. Calculations of 50 year financial commitments often involve 360 dimensional integrals. With all the bad news regarding the exponential rise of required computational resources for dealing with such integrals, there is a Rosetta Stone which over the past 50 years has indicated there is hope for us in connecting with the realm of higher dimensions. This is the Monte Carlo method, where points in the domain of integration are chosen randomly with respect to the integration measure. While worst-case results indicate computational complexities for ϵ-approximations of integrals which rise on the order of $n(\epsilon) \sim \left(\frac{1}{\epsilon}\right)^{d/r}$(with d dimension and r the Sobolev regularity of the function), Monte Carlo gives us essentially $n(\epsilon) \sim \left(\frac{1}{\epsilon}\right)^2$. There is no free lunch, and the price paid is that this complexity is actually an average over all choices of the n random

points $\{x_i\}_i$ at which $f(x)$ is to be evaluated, with no worst case guarantees. Practically, this dimension independence of Monte Carlo says there is hope, and work by Wozniakowski and others has shown that there are *deterministic* choices of x_i which work essentially as well, though only when averages are then taken over the function space F under Gaussian probability measures (and presumably other broader classes; see also [SW]).

A more formidable extension of high-dimensional integration is integration in infinite dimension. Consider a path integral, say over a Gaussian measure on a Hilbert or Banach function space H, as occurs in particle physics, statistical mechanics, solutions of PDE, and mathematical finance. How can we compute such an integral? Given a function f *on* H, we wish to compute

$$\int_H f \, d\mu(f), \tag{1}$$

with respect to a measure μ on H. For functions with finite regularity r (belonging to a Sobolev space of order r), the problem of evaluating (1) with error ϵ or less is effectively intractable in the worst case, except (in the case of a Gaussian measure μ) in trivial cases where the rank of the covariance operator C_μ of μ is finite. Monte Carlo, however, stays true to form (as might be expected from its dimension independent complexity), and yields complexity of order ϵ^{-2}. Thus, $O(\epsilon^{-2})$ elements of real-number information need to be obtained about f in order to obtain error of order ϵ in approximating (1) (see also [Wo]). Some of the true power of Monte Carlo is exemplified in infinite dimensions, including so-called quantum Monte Carlo methods for evaluating eigenfunctions of Schrödinger operators [CA].

A more formal framing of the complexity issues is as follows. Computational input for a problem consists of partial information Nf about an object f (possibly a function). The presumed generality of function spaces allows us to assume $f \in F$, a collection of functions known a priori to contain f. For example F might be a Sobolev space of smooth functions, with a restriction on smoothness of the form $\|f\|_F \leq K$. Such assumptions constitute *a priori* information about f. The information, e.g., $Nf = (f(x_1), f(x_2), \ldots, f(x_n))$ (*standard information*), or more generally

$$Nf = (L_1 f, \ldots, L_n f)$$

is *a posteriori* information.

Many input-output tasks, including perceptual ones (which computers now try to model), require a judicious combination of the two types of information. To give a seemingly pedestrian example from everyday life, if one sees a black shape move across one's field of vision and knows *a priori* there is a dog in the house, one might add information very quickly to strictly perceptual data by adding a head and legs to a vaguely seen image. Along with potential neural networks of the future, the brain itself can be viewed as a Bayesian machine, which from incredibly sparse information Nf can produce remarkably detailed and accurate conclusions Sf.

In approximation theory a common *a priori* assumption about functions f is smoothness, which results in spline approximations as optimal interpolants of standard information Nf. *A priori* assumptions are often implicitly made, but sometimes it is important to state them, so the best combinations of the two types of knowledge can be made. A future challenge might be the study of other types of assumptions and their usefulness in extracting better approximations from sparse information.

The book is organized as follows. In Chapter 2 there is a brief introduction involving a standard motivating example for the partial information model used in information-based complexity theory. This involves integration of a function f on $[0,1]$, and the question of what choices of points $\{x_i\}_{i=1}^n \subset [0,1]$ and constants $\{c_i\}_{i=1}^n$ are optimal for quadrature, i.e., the approximation

$$\int_0^1 f(x)dx \approx \sum_{i=1}^n c_i f(x_i) \tag{2}$$

(the choices of $\{c_i\}$ from the trapezoidal and Simpson's rules come to mind most easily, though these do not constitute optimal algorithms). The information aspect of this procedure is clear, with standard information

$$Nf = (f(x_1), \ldots, f(x_n)).$$

The algorithm ϕ consists of computation of the sum (2). Examples of other aspects of the separation of information and algorithm operations follow after this.

Chapters 3, 4, and 5 address questions of tractability discussed above, from large finite dimensional problems (such as those in mathematical finance), to infinite dimensional ones (path integration). There are computer experiments which show that quasi-Monte Carlo methods (which are deterministic) beat Monte Carlo by factors of ten to a thousand for problems of mathematical finance. A possible explanation due to Sloan and Wozniakowski [SW] is presented.

Chapter 7 is an introduction to nonlinear problems. The occasional frustration of workers who have dealt with these is summarized in the book's observation: *nonlinearity is not a property - it is the lack of a property.* Nonlinear problems come in a menagerie of forms, and so must often be considered individually. A quintessential problem appearing in numerous mathematical applications is also the quintessential nonlinear problem - optimization. There have been some inroads into this by many, including some workers in continuous complexity, though the field still sometimes seems virtually undented.

A discussion of models of computation follows. The issue of real number versus bit computational models is summarized in the question, *what should we do when a continuous mathematical model meets a finite state machine?* Where the transition from infinite to finite dimension occurs determines the model of computation.

Chapter 9 involves a comparison of intractability and undecidability. Just as Gödel's undecidability has placed limitations on what is provable and eventually knowable, the authors argue convincingly that intractability has the potential to do the same thing. For example, if the computation of a number x has a lower bound of 10^{10^i} operations for the computation of the i^{th} digit, then effectively this number is unknowable within the present model of our universe. Do such results have implications as fundamental as Gödel's? It is not known, but cannot be ruled out (see also, e.g., [T2]). The effective question is, can intractability limit scientific knowledge?

Following this are chapters on mathematical analyses of complexity of linear programming, verification, implementation testing, and computations in the presence of noisy information. At the end of the book, some topics are covered which stimulate the imagination, including paradigms which assign numerical values to mathematical hypotheses, and values to information in computation. The latter are related to comparison of *a priori* and *a posteriori* uncertainties in mathematical problems. The book concludes with a recap of open problems and a very brief history of information-based complexity theory, along with an extensive bibliography.

This book has the nature of an overview of the area, with a discussion of "hot" and inspiring current applications. For more concrete functional analytic foundations, reviews of past work, and other overview issues, we refer to [PW], [PT], [TW], [TWW].

Overall this book is a good short introduction to a field which has found interesting applications such as mathematical finance, and deserves a current treatment. The approach is philosophical, and so gives a good grounding in the essential issues of complexity and information. The book does not pretend to be all-encompassing, and discusses only a few of the said thirty definitions of complexity which have been in use [L]. Issues of computational tractability come in many forms, and the real-number model of computation is a very important one to consider. This reviewer likes overviews and approaches to thinking about mathematical and practical issues on various levels. For an overview of the larger issues, this small book is excellent.

References

[BSS] L. Blum, M. Shub, and S. Smale, On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions, and universal machines, *Bull. Amer. Math. Soc.* **21** (1989), 1-46.

[BCSS] L. Blum, F. Cucker, F. Shub, and S. Smale, *Complexity and Real Computation*, Springer-Verlag, New York, 1998.

[CA] D. Ceperley and B. Adler, Quantum Monte Carlo, *Science* **231** (1986), 555-560.

[HS] M. Hirsch and S. Smale, On algorithms for solving $f(x) = 0$, *Comm. Pure Appl. Math.* **2** (1979), 281-312.

[L] Lloyd, S, *Measures of Complexity,* Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA.

[PW] E. Packel and H. Woźniakowski, Recent developments in information-based complexity, *Bull. Amer. Math. Soc.* **17** (1987), 9-36.

[PT] E. Packel and J. Traub, Information-based complexity, *Nature* **327** (1987), 29-33.

[S] S. Smale, On the efficiency of algorithms of analysis, *Bull. Amer. Math. Soc.* **13** (1985), 87-121.

[SW] I. Sloan and H. Woźniakowski, When are quasi-Monte Carlo algorithms efficient for high dimensional integrals? *J. Complexity* **14** (1998), 1-33.

[T1] J. Traub, *Iterative methods for the solution of equations*, Prentice-Hall, Englewood Cliffs, N.J., 1964.

[T2] A Continuous Model of Computation, *Physics Today*, May 1999, 39-43.

[Wo] H. Woźniakowski, Complexity of multivariate problems with applications to path integrals, *Z. Angew. Math. Mech.* **3** (1996), 131-134.

[TW] J. Traub and H. Woźniakowski, Breaking intractability, *Scientific American* **270** (1994), 102-107.

[TWW] J. Traub, G. Wasilkowski, and H. Woźniakowski, *Information-Based Complexity*, Academic Press, Boston, 1988.

[W] A. Werschulz, *The Computational Complexity of Differential and Integral Equations: An Information-Based Approach*, Oxford University Press, New York, 1991.