In this section, the IEEE Control Systems Society publishes reviews of books in the control field and related areas. Readers are invited to send comments on these reviews for possible publication in the Technical Notes and Correspondence section of this Transactions. The CSS does not necessarily endorse the opinions of the reviewers.

If you have used an interesting book for a course or as a personal reference, we encourage you to share your insights with our readers by writing a review for possible publication in the Transactions. All material published in the Transactions is reviewed prior to publication. Submit a completed review or a proposal for a review to:

D. S. Naidu
Associate Editor—Book Reviews
College of Engineering
Idaho State University
833 South Eighth Street
Pocatello, ID 83209 USA


This is a book every modern control engineer should have on his/her reference bookshelf. It is worth the $89.00 price even if you have to pay for it out of your personal funds. First, it collects in one place, in consistent notation, all of the information on computational Intelligence (CI), such as Neural Networks (NN), Fuzzy Logic (FL), Genetic Algorithms (GA), and other acronyms like SA, radial basis function networks (RBFN’s), etc., that you always wanted to know but were afraid to ask regarding the mass of jargons that have grown over the years in the subject. Second, this is a thoroughly modern book complete with Matlab exercises, companion floppy disks (for the asking from the publisher), and websites for the latest updates and resources. Third, the book is remarkably restrained in its hype for the subject, which cannot be said for many other works in this area. Finally, there is useful information in here that control engineers should know beyond the traditional tools of the trade such as $H^\infty$, robustness, LQG, Lyapunov, and other differential equation/frequency-domain-based techniques.

Having said this, let me provide a guided tour for the readers and some reservations.1 Before doing this, however, we shall first state some well-known facts and self-evident truths (at least to the readers of the IEEE Transactions on Automatic Control) with which to better understand the structure of CI and to penetrate the jargons.2

I. CONVERTING DYNAMIC OPTIMIZATION PROBLEMS TO STATIC ONES

The principal idea in dynamic optimization is to convert long-term effects into short-term consideration. In dynamic programming one does this through the use of the optimal cost-to-go function $V$ (state). In optimal control, it is the LâGrange multiplier function $\lambda(t)$. Both devices summarize the future effect of present control in terms of its immediate effect on the next state. This way a dynamic optimization problem can be reduced to a series of static optimization problems. The price you pay is of course that you must solve either the functional equation or the two-point boundary value problem of optimal control that links the series of static problems.

II. ITERATION TRADEOFF BETWEEN ESTIMATION AND UPDATING

When you are trying to solve a stochastic optimization problem, $\min_x \{ \max_{\theta} E_x [ L(\theta, \xi) ] \}$, or the equivalent root finding problem, $\nabla_x E_x [ L(\theta, \xi) ] = 0$, via successive approximation, you face the following tradeoff. Since the value of any expectation, $E[L]$, must be estimated, one must decide on how accurately to estimate the quantity before using it for the purpose of updating or hill climbing. For a finite computing budget, one can either spend most of the budget on estimating each iterative step very accurately and settle for fewer steps, or estimate each step very poorly but use the budget to calculate many iterative steps. In the former, you have accurate information but have few chances of improvement, while in the latter you have poorer information but can afford many chances for improvement. Particularly in dynamic system optimization problems, one has the issue of trading updating (hill climbing) a strategy/policy/control law after every state transition, or postponing the updating until the entire trajectory based on the policy is played out, or anywhere in between. The choice of the step length parameter in the Robbins–Monroe stochastic approximation algorithm is another quantitative manifestation of this tradeoff consideration for convergence purposes.

III. STATISTICAL FUNCTION FITTING

The generic situation is that we have a mass of data and we wish to summarize it using a model with much fewer parameters. Once thus modeled, we can economically use the model in place of the original data and/or generate new data which we hope will have the same statistical properties as the original (the so-called generalization property). The simplest version of this process is linear regression,

1 We should point out that solving the Bellman equation in stochastic dynamic programming is a problem of this type.

Publisher Item Identifier S 0018–9286(97)06595-1.

1 Or rather urges to the authors for their next edition.

2 The reviewer hastens to point out that the authors do a very good job of defining and explaining the jargon. However, they have not provided the control theory perspective that Transactions readers might like to have in understanding CI.
where you use a linear parametric model to least square fit a set of
given data. The fit is declared complete if the “residue” can be
identified as i.i.d. noises. This is in the sense of all “information”
having been squeezed out of the data, and there is no way to further
predict the white noise residue. More generally, nonlinear models can
be used for the fit. And if the black box generating the data is known
to have a certain structure, then the same structure should be adopted
in the fitting model. This is the basis of the Kalman filter which in
essence says that using polynomials to fit general dynamic systems
output is inefficient and plainly wrong. We should use the system
responses themselves as the fitting functions. This is why we see the
presence of both the system dynamic equation and the observation
equation in the filter. On the other hand, if nothing is known about
the source generating the data, then a general model-free architecture
is adopted. There are many generic “model free” candidates for data
fitting starting with polynomials, Fourier series, to wavelets, RBFN’s,
and other NN’s, each with their advantages and disadvantages. For
example, wavelets and certain neural networks, unlike the Fourier
series or threshold functions, have the property that the response
of individual units/nodes are localized and drop to zero as inputs
approach infinity. This property can be important for many modern
applications.

IV. COMBINATORIAL EXPLOSION OF STATE SPACE

Many problems/techniques are conceptually simple and elegant to
explain and execute, but they do not scale up well, i.e., the computa-
tions involved grow exponentially with the size of the problem.
This renders such techniques impractical. The entire spectrum of
NP-hard combinatorial optimization problems, such as the traveling
salesman problem, belong in this category. In our realm, this difficulty
often crops up as the size of the state space or the “curse of
dimensionality.”

In fact, without this fundamental difficulty, dynamic
programming would have declared the end of control theory research
way back in the 1960’s. In the writing of academic papers, we tend
to ignore or sidestep this fundamental but unpleasant fact. However,
awareness of it helps to explain why certain research directions are
taken and others not and to assess what technique is practical and
what is merely conceptual.

Now we can review the book in terms of these four fundamental
ideas described above. We shall use shorthand notations, dynamic
to static conversion (D → S), estimating versus updating (EvstU),
statistical function fitting (SFF), and combinatorial explosion (CE),
respectively, to reference these four ideas and the concepts behind
them.

The book begins with an excellent introductory chapter explaining
what is CI and its difference and relationship to artificial intelligence
(AI). The next three chapters (Chs. 2–4) constituting Part I of the
book deal with FL. We shall postpone its discussion until later. Part
II (Chs. 5–7) succinctly reviews techniques of real variable-based
optimization, in particular, least square fit, iterative hill climbing
using gradients, and derivative-free optimization searches including
genetic algorithm and simulated annealing commonly associated with
discrete or structureless optimization problems.

This material is immediately put to use in Part III (Chs. 8–11) on
NN’s and learning. The chapters are well written and comprehensive.
This reviewer found that the material is best approached from the
following perspective. “Learning” is the root of all intelligence,
natural or artificial. We used the word in the sense of the paradigm:

\[ \text{DATA} \Rightarrow \text{INFORMATION} \Rightarrow \text{KNOWLEDGE} \]

5 In the book and in the literature, reinforcement learning is particularly
associated with learning in a dynamic setting.

6 In learning to design a linear/nonlinear filter, the criterion is to reduce the
estimation error residue to i.i.d. white noise.

We wish to thank W. Gong for pointing out that the problem of CE some-
time arises by way of complex constraints. Contributions of the algorithms
of linear and integer programming are due to the way they efficiently handle
constraints. For the same reason, uniform sampling in a general irregular
region can be very difficult practically.

6 A principal issue is a more involved identi-
fication of cause-and-effect in the sense that future desired output
now depends on all past input. This is sometimes referred to as
the credit assignment problem, i.e., which past input deserves credit
for the success in current output. This is where the idea D → S
enters. Dynamic programming is invoked to convert the problem
to a series of static learning problems. The price one pays is the
determination of the optimal cost-to-go function, V (state), and the
solution of the Bellman equation. In general, of course, the Bellman
equation cannot be solved in closed form, and iterative computations
are required. Here the idea of estimation versus updating (EvstU)
the V function comes into play. The techniques of policy iteration
and TD(λ) discussed in Chapter 10 are simply different tradeoffs of
EvstU when attempting to determine the optimal cost-to-go function
V (state) by solving the Bellman equation. Q-learning is also another
related approach to the solution of the Bellman equation. Finally,
because of the CE difficulty, instead of trying to solve V (state) for
each state, we attempt to use neural networking as SFF to determine
V approximately. This is the rationale behind the neuro-dynamic
programming method of Bertsekas–Tsitsiklis.

Having reduced the problem of dynamic constraints into static
problems, we can concentrate on the static version of the problem
of learning. It is here multilayered-perceptron or neural networks
(MLP/NN) found their roles. Chapters 8 and 9 explain the basics
of MLP/NN as model-free SFF in the sense described above. Since
SFF requires the solution of an optimization (often least square fit)
problem, techniques of Part II are used effectively here to explain
learning with a teacher, back propagation, hybrid learning, and mixed
linear and nonlinear least square fit. Finally, unsupervised learning in
Chapter 11 can be considered as SFF optimization problems with
built-in or implicit teachers based on similarity or other criterion.

RBFN’s and Hopfield networks are simply additional families of
fitting functions used in these settings, although the latter can be
viewed as SFF only in a general sense.

Returning to Part I (Chs. 2–4) on FL, we find something new, i.e.,
outside the fundamental ideas 1–4 above. Over the years Zadeh has
persuasively maintained that interactions with the human-made word
ultimately must be done via human languages which are imprecise
and fuzzy. One way or another, analysis cannot proceed without
some translation between the human and the mathematical languages.
For example, to capture the experience-based control rule “if the
temperature is HIGH and the pressure is LOW, then turn up knob
K by a SMALL amount,” we must define what we mean by the

4 We wish to thank W. Gong for pointing out that the problem of CE some-
time arises by way of complex constraints. Contributions of the algorithms
of linear and integer programming are due to the way they efficiently handle
constraints. For the same reason, uniform sampling in a general irregular
region can be very difficult practically.
capitalized words. FL provides a systematic and quantitative way of doing this versus ad hoc approximations as practices in the art of engineering. In traditional system analysis, we are also principally concerned with a mapping \( y = f(x) \), where \( f \) may represent a very complex composition of functions. In a sentence, Chapters 2–4 deal with the same issues when \( x, y \), and more importantly \( f \) may be given in human language terms as well as in mathematical languages (e.g., the if–then rule stated above in English). It is this translation ability provided by FL that enables the systematic incorporation of human language-based experience into learning techniques of Part III reviewed above. This important contribution brings us directly into Part IV (Chs. 12 and 13) and Part V (Chs. 14–16).

FL also enables us to capture human experience quantitatively and systematically into learning models. In particular, we can divide model structures or architectures from various fuzzy if–then rules or human language-based descriptions of the systems. Such “knowledge” helps learning immensely versus the model-free or black box learning approaches of Part III. The ANFIS and CANFIS architecture in this part demonstrate the advantage of incorporating this structural knowledge. Chapters 14–16 also address some issues not commonly encountered in continuous variable system modeling, namely, the identification of structure in discrete sets. The pruning of regression trees and structure identification in rule-bases have their parallels in the datamining literature. The central theme here is the problem of overfitting of data or how to be parsimonious with respect to parameters in SFF.

Parts VI and VII finally deal with the application of the ideas discussed above to case studies involving the control of dynamic systems, pattern recognition, game playing, and complex industrial problems. It is here that this reviewer finds the promises of CI not completely fulfilled. This is not particularly the fault of the authors but rather an indication of the relative youth of the field which is sometimes obscured by the enthusiasm of people working in it. In fact, to the authors’ credit they are honest about the shortcomings of the state-of-the-art of CI in the book (e.g., p. 434 on knowledge acquisition). In the chapters of neuro-fuzzy control and on advanced application, the original challenge of parking a car advanced by Zadeh at the start of FL research some 32 years ago remains untouched. Except for the part played by the translation ability of FL, it is not entirely convincing to this reviewer that the problems could not have been solved by traditional techniques. Whether it is the “knowledge” about the problem or the power of the techniques of CI that contributed more to the solution of the application cases is less clear. In fact, the importance of knowledge is made abundantly evident when in Chapter 22 the straightforward application of model free MLP and genetic algorithms without structural knowledge resulted in inferior performances. Knowledge acquisition is of course the $64$ million question in AI or CI. We know relatively little about how-to-capture and what-is knowledge.

It is not the fault of the book that this problem remains unsolved. Lastly, this reviewer feels that insufficient attention or emphasis has been paid to the issue of CE. Not all CI techniques scale up well. And the reader should not read more into the techniques than there is. Just to mention one particular example, GA requires the evaluation of the “fitness” of a population of possible alternatives. It is routinely assumed that in the implementation of GA this fitness can be easily evaluated. However, in many real world problems, this fitness evaluation can only be “estimated” via a computationally intensive Monte Carlo simulation, e.g., the performance of a complex semiconductor manufacturing plant, or a multi-service military logistic supply chain under a given operational strategy. In such cases, one iteration of a GA may impose an infeasible computational burden on the algorithm. Such a computational burden faces the fundamental limit of simulation and cannot be improved. Such warnings are not often given or emphasized in academic papers and books.

In fact, it is the opinion of this reviewer that the resolution of the CE problem is intimately related to the problem of knowledge acquisition and heuristics. NP-hardness is a fundamental limitation on what computation can do. Quantifying heuristics and acquiring structural knowledge seems to be the only salvation for the effective solution of complex real problems. In fact, these are the reasons for the continuing existence of human researchers and problem solvers.

Despite these reservations, this reviewer urges the control community to embrace the subject of CI and recommends this book. CI is mind broadening and opens up a host of real world problems beyond those differential equation-based problems that this Transactions has lived with for the past 35 years. The methodologies are not hard to learn (in fact quite familiar once you get beyond the jargons and acronyms) when viewed in terms of the perspective of ideas 1–4 above. More importantly, readers of this Transactions have a great deal to contribute because of such perspectives. While there are numerous fine books on the subject of fuzzy logic, genetic algorithms, and neural networks individually, this book puts all the subjects together in one place and shows their respective places in CI. The reviewer congratulates the authors for this timely book.

ACKNOWLEDGMENT

The reviewer wishes to thank students in the DEDS group at Harvard, E. Lau, L.-H. Lee, J. Lee, D. Li, and M. Yang, for studying and discussing the book under review. They helped to clarify many issues. However, the reviewer alone is responsible for the opinions expressed.


9However, this reviewer is not aware that the parking problem has been treated anywhere else including nonfuzzy control theory. An easier version of the problem, backing a truck to a loading dock, was treated by Sastry et al.

10This reviewer does concede that human-based knowledge can be more easily and systematically captured via FL than with the traditional “art of engineering practice.” This, after all is the contribution of FL and the CI community. CI and FL also seem to offer a better impedance match to complex system problems than some the rarified theories of systems control.
Direct link J.-S.R.Jang, et al - Neuro-fuzzy and soft computing.. a computational approach to learning and machine intelligence.djvu
Download, download J.-S.R.Jang, et al - Neuro-fuzzy and soft computing.. a computational approach to learning and machine intelligence.djvu 4shared for all, at: 2009-02-27T13:39:36.000Z. Minimal ADB and Fastboot Tool helps in installing ADB and Fastboot drivers on your windows machine. So, that you can use ADB or Fastboot commands to perform some tasks like adb sideload, installing TWRP Recovery, flashing etc. Read article. Pvr IPTV Simple Client Download. IPTV Simple PVR Client support m3u playlists, streaming of Live TV for multicast/unicast sources, listening to Radio channels and EPG. Read article. Exiso Gui 1.4 b Download. Neuro-Fuzzy and Soft Computing-A Computational Approach to Learning and Machine Intelligence [Book Review]. Article (PDF Available) in IEEE Transactions on Automatic Control 42(10):1482 - 1484 Â· November 1997 with 4,223 Reads. DOI: 10.1109/TAC.1997.633847 Â· Source: IEEE Xplore. Å To send comments on these reviews for possible publication in the Technical Notes and Correspondence section of this T. RANSACTIONS. The CSS does not necessarily endorse the opinions of the reviewers. Å Pocatello, ID 83209 USA. Neuro-Fuzzy and Soft Computing—A Computational Approach to Learning and Machine Intelligence—J. S. R. Jang, C. T. Sun, and E. Mizutani (Englewood Cliffs, NJ: Prentice-Hall Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence. PreKâ€“12 Education. Higher Education. Å The book is oriented toward methodologies that are likely to be of practical use; many step-by-step examples are included to complement explanations in the text. Pg.___ Specially designed figures provide a visual reinforcement for as many ideas and concepts as possible. These figures were generated using MATLAB and these MATLAB files are available via FTP or WWW. Pg.___ Includes exercises, some of which involve MATLAB programming tasks which can be expanded into suitable term projects. This will provide the student with hands-on programming experiences for practical problem-solving. Pg.___