Pergamon

0022-4073(93)E0030-V

J. Quant. Spectrosc. Radiat. Transfer Vol. 51, No. 1/2, pp. 247–253, 1994

0022-4073/94 $6.00 + 0.00

THE USE OF ARTIFICIAL NEURAL NETWORKS IN PLASMA SPECTROSCOPY

W. L. MORGAN, J. T. LARSEN, and W. H. GOLDSTEIN

Kinema Research, P.O. Box 1147, Monument, CO 80132, Cascade Applied Sciences, Inc., P.O. Box 4477, Boulder, CO 80132, and Lawrence Livermore National Laboratory, Livermore, CA 94550, U.S.A.

Abstract—Plasma spectroscopic instrumentation produces a vast amount of data that requires a substantial effort in analysis. New methods are needed to improve the efficiency of the analysis process. Artificial neural networks, which have been applied to a variety of signal processing and pattern recognition problems, are well suited for the analysis of laboratory data that can be viewed as a pattern matching problem. We summarize the neural network concepts and algorithms and illustrate the use of a neural network in the determination of electron temperature and density by analysis of spectral data from a plasma.

INTRODUCTION

The standard technique for finding the temperature and density of plasmas, such as those found in astrophysical objects, in laboratory magnetically confined or inertially confined fusion plasmas, and even in laboratory gas discharges, has long been spectroscopy.[1,2] One measures the peak intensities, and perhaps the widths, of a number of spectral lines being emitted by the plasma and compares the results with model calculations in order to deduce T_e and N_e, the electron temperature and density. The technique is obviously strongly model dependent but there usually is no alternative. Typically, certain spectral lines will be brighter at certain temperatures and densities and other lines will dominate in other regions of T_e and N_e. The general principle is that measurement of enough lines will give a unique estimate of temperature and density. The models may be fairly simple, such as the limiting cases of local thermodynamic equilibrium or coronal equilibrium, or they may be quite complex collisional radiative models possibly even including hydrodynamics. The process of modifying parameters in such models; performing the collisional radiative calculations; generating synthetic spectra; and comparing with the measured spectra takes considerable effort and time.

The conventional modeling process[3,4] used to extract plasma conditions and charge state abundances consists of solving the collisional radiative rate equations; generating synthetic spectra; and fitting to a measured spectrum by minimizing a function such as

$$\chi^2 = \sum_i [(I_i^e - I_i^m)/\Delta I_i^e]^2$$

where the sum is over the intensities of lines i and the e and m denote the experimentally measured and the model quantities. The collisional radiative models typically incorporate a detailed atomic physics model for important low lying ionic states and an average model for higher lying levels. The electron impact and photon processes among the levels and between the bound levels and the continuum are dealt with in varying degrees of sophistication. There are, in addition, assumptions that are typically made about the time dependence of the kinetics of the ionic energy levels versus the time dependence of the ionization balance. A typical assumption is a quasi-steady-state model[4] that allows a non-equilibrium ionization balance, but assumes the excited level populations to be in steady-state due to the short timescale for level equilibration.[5–7] Of course, the more sophisticated the model is, the more time and effort consuming is the process of extracting information on the state of the plasma.

In this paper we discuss how artificial neural networks that have been trained on detailed

collisional radiative models might be used in spectroscopic analysis of plasmas. The concept that we are exploring is that this approach to plasma spectroscopy may potentially be able to simplify and speed up the analysis process.

ARTIFICIAL NEURAL NETWORKS

The concept of artificial neural network[8–10] is based on our understanding of how human brains function and how we learn and are able to generalize what we learn. For example, we can learn enough from observation of a finite set of experiments to attempt to predict the result of a new, but not too different, experiment. We could say that given a set of observations of the output vector $\mathbf{y}$ for a given input vector $\mathbf{x}$ in a finite series of n experiments:

$$(\mathbf{x}_1,\mathbf{y}_1),\ (\mathbf{x}_2,\mathbf{y}_2),\ \ldots,(\mathbf{x}_n,\mathbf{y}_n)$$

we have learned enough about the unknown functional relationship $\mathbf{y}=f(\mathbf{x})$ to predict $\mathbf{y}$ when presented with a new $\mathbf{x}$ not belonging to the known set. This kind of learning in order to develop a predictive capability is what our brains do and what artificial neural networks are designed to do.

Based on notions of how human brains are structured, artificial neural networks consist of layers of simulated "neurons" with associated activation functions, transfer functions, and weighting functions for the "synapse" connections to other neurons. As shown in Fig. 1, the key elements are an input layer, one or more "hidden" layers, and an output layer. Each neuron has a transfer or response function associated with it, typically the logistic function $T(x) = 1/(1+\mathrm{e}^{-x})$, that gives an output value that is a non-linear function of the sum of the input values. The input values are the weighted outputs of each neuron in the previous layer. That is, if the output of neuron j is o_j and w_{ij} is the weight connecting neurons i and j, then the output of neuron i is:

$$o_i = T\left(\sum_j w_{ij}o_j\right) = 1/(1+\mathrm{e}^{-\sum_j w_{ij}o_j})$$

where the sum is over all neurons j having outputs that feed into neuron i. The hidden layers give the network a high degree of non-linearity and, from the point of view of the network as a pattern matcher, provide an internal representation of the correlation between the input and the output patterns. The concept behind this kind of network (known as a *feed-forward, back-propagation* network) is that it can "learn" to associate a set of output patterns with a set of input patterns by modifying (strengthening or weakening) the weights associated with the connections between pairs of neurons in response to the network's successes and failures so as to optimize in favor of

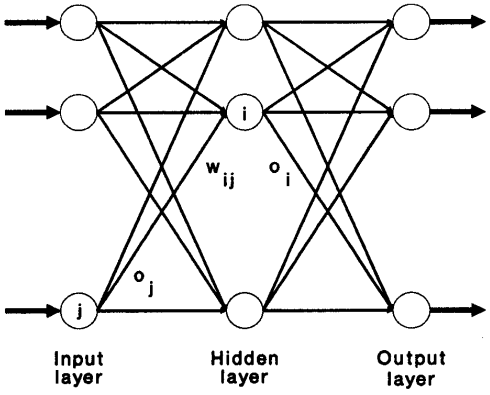

Fig. 1. Three layer feedforward, backpropagation neural network. The neurons (○) in a given layer operate via a transfer function on the weighted output from the neurons in the previous layer to produce an output signal that is passed on to the next layer.

the network's successful strategies. The network is trained by running a number of cases of known {input,target output} sets through it and adjusting the weights to minimize the sum of the squares of the differences between the desired result and the computed result. The neural network will assimilate and correlate data given it in the process of "training" so that when given input vectors outside of the training set it will produce reasonable values for the corresponding output vectors. The matrix of weights, w_{ij}, represents the mapping between the set of input vectors and the set of output vectors and contains all the information correlating the output to the input. One might also think of this kind of neural network as a highly flexible many parameter highly non-linear interpolation algorithm. It has an advantage over many standard algorithms in that neural networks have a generalization ability and are able to deal with noisy or even missing data.

The network is trained by running a number of cases of known {input,output} sets through it and adjusting the weights to minimize the sum of the squares of the differences between the desired result and the computed result. This quadratic function is the so-called *energy*, *cost*, or *objective* function. The weights are adjusted using what is known as the generalized delta rule.[8–10] The energy function, E, can be written

$$E = \sum_p \sum_i [t_i(p) - o_i(p)]^2,$$

where $t_i(p)$ is the value of the training output for output neuron i and training data set p; $o_i(p)$ is the corresponding network output. E is minimized using an iterative procedure whereby the weights w_{ij} are adjusted according to:

$$\Delta w_{ij} = \sum_p \epsilon \delta_i(p) o_j(p),$$

where

$$\delta_i(p) = [t_i(p) - o_i(p)] o_i(p)[1 - o_i(p)]$$

for output layer neurons, and

$$\delta_i(p) = o_i(p)[1 - o_i(p)] \sum_j w_{ij} \delta_j(p)$$

for hidden layer neurons. The parameter ϵ is called the training rate coefficient. The weights are adjusted starting with the neurons in the output layer and moving back a layer at a time toward the input layer.

The neural network will assimilate and correlate data given it in the process of "training" so that when given input vectors outside of the training set it will produce reasonable values for the corresponding output vectors. All the information correlating the output to the input is contained in the set of weights for a given network structure.

NEURAL NETWORKS FOR PLASMA SPECTROSCOPIC ANALYSIS

We have been developing a neural network for use in the plasma x-ray spectroscopic analysis of high density laser produced plasmas. A preliminary report on this work appeared in Ref. 11. In developing the neural network techniques for this application we considered the K-shell spectra of a laser produced aluminum plasma. These spectra consist of a number of lines that are emitted by hydrogen-like and helium-like aluminum ions in the plasma. Measurements of spectra of such plasmas can be found in Refs. 4 and 12. The spectral lines that we are interested in and for which we have measurements are the H_α, H_β, H_γ, H_δ, He_α, He_β, He_γ, He_δ, and He_{ic}. This last is the so-called helium intercombination line, which is the $1s2p\ (^3P_1) \rightarrow 1s^2(^1S_0)$ transition in He-like Al. We use relative intensities as input to the neural network; the network consequently has nine input neurons. The output neurons are T_e, N_e, and the fraction of helium-like aluminum. The neural network that we are currently working with has two hidden layers each having 15 neurons.

We have used the plasma modeling program RATION[3] to generate the synthetic data used in training the neural network. Using RATION we computed 192 training patterns spanning the space of $50 \leqslant T_e \leqslant 1200$ eV, $5 \times 10^{19} \leqslant N_e \leqslant 6.4 \times 10^{21}$ cm^{-3}, and $1.6 \times 10^{17} \geqslant N[\mathrm{He}]/N[\mathrm{H}] \geqslant 6.5 \times 10^{-2}$.

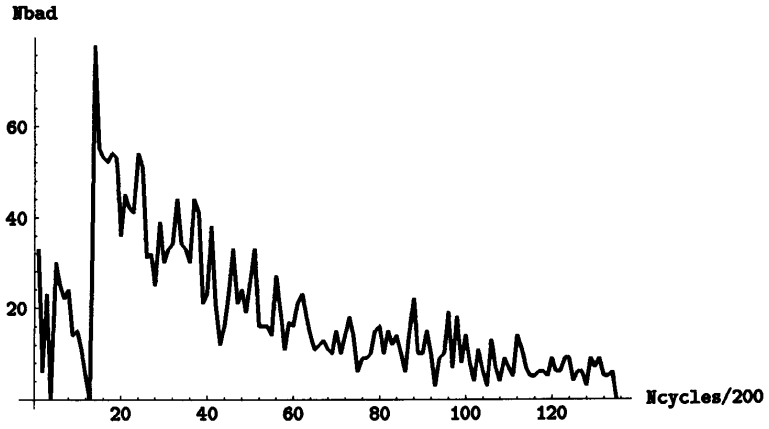

Fig. 2. The number of patterns for which the neural network failed to match the target output as a function of the number of passes (Ncycles).

The network had difficulty in dealing with 19 order of magnitude range in the ionization balance, so within the program this is converted to the He-like fraction, as noted above. The network has been trained on the synthetic data to a tolerance of 5%.

Figure 2 shows a summary of the training of the network to the 10% level. First the training criterion was set at 30% with $\eta_o = 0.9$, $\eta_h = 0.5$, and $\alpha = 0.9$. It took 800 cycles, i.e. passes of the full set of training patterns through the network, to converge to the 30% level. The criterion was then set to 20% with the training parameters the same and the network then converged to this error value in 1694 cycles after the restart. The error criterion was then set to 10% and the network was restarted with the same training parameters for another 10^4 cycles. It was then run for another 10^4 cycles with $\eta_o = 0.5$, $\eta_h = 0.3$, and $\alpha = 0.5$, and then for another 4309 cycles with the training parameters being 0.2, 0.1, and 0.2 respectively until it converged. The progress of training can be seen in Fig. 2. Most of the difficulty in training had to do with the second neuron, i.e. N_e. This probably indicates an insensitivity of the spectra to N_e compared to T_e and fractional ionization.

This network was eventually trained to the 5% level. The average error in T_e and ionization balance was about 1% with maximum errors of about 4%. The histogram in Fig. 3 shows the distribution of errors for the 192 training patterns for the second output neuron, i.e. N_e. The average is 1.8% and the maximum is just about 5%. T_e and fractional ionization are trained to much better accuracy.

ANALYSIS OF TOKAMAK USING THE NEURAL NETWORK

Plasmas produced in tokamaks are amenable to x-ray spectroscopy since near steady state equilibrium can be obtained for several hundreds of milliseconds and spatial homogeneity can be approached near the plasma axis. Experiments with small concentrations of trace elements have been used for several years to measure the plasma electron density and temperature in tokamaks.

One such experiment involved seeding the Alcator C tokamak plasma with a 0.1% concentration $N[\mathrm{Ar}]/n_e$ of argon gas. The resulting helium-like spectrum consists principally of resonance, intercombination, and forbidden lines (magnetic dipole transition) together with $n = 2$ satellites and some $n \geqslant 3$ satellites at the resonance line.[13]

Figure 4 shows the detail of a typical He-like Ar spectrum, where the indicated transitions are:

Key	*Transitions*
w	$1s^2\,{}^1S_0$–$1s2p\,{}^1P_1$
x	$1s^2\,{}^1S_0$–$1s2p\,{}^3P_2$

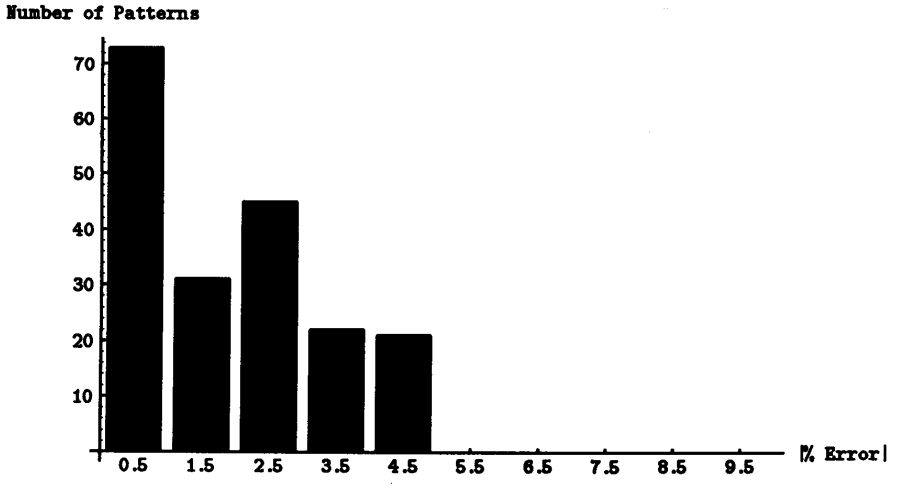

Fig. 3. Histogram of the fractional difference between the network output and the training target output for N_e.

y $1s^2\,{}^1S_0$–$1s2p\,{}^3P_1$
z $1s^2\,{}^1S_0$–$1s2s\,{}^3S_1$
q $1s\,{}^22s\,{}^2S_{1/2}$–$1s\,(2s2p\,{}^1P)\,{}^2P_{3/2}$
r $1s\,{}^22s\,{}^2S_{1/2}$–$1s\,(2s2p\,{}^1P)\,{}^2P_{1/2}$
k $1s\,{}^22p\,{}^2P_{1/2}$–$1s2p^2\,{}^2D_{3/2}$

In a high temperature plasma, the states are mainly populated through electron impact excitation from the 1^1S_0 ground state with cascade contributions from higher excited states. Owing to the long lifetime of the 2^3S_1 state, the collisional excitation to the 2^3P_j state, with a rate proportional to the electron density, must also be considered. The presence of hydrogen-like and lithium-like charge states in the plasma gives rise to recombination and ionization contributions in proportion to the

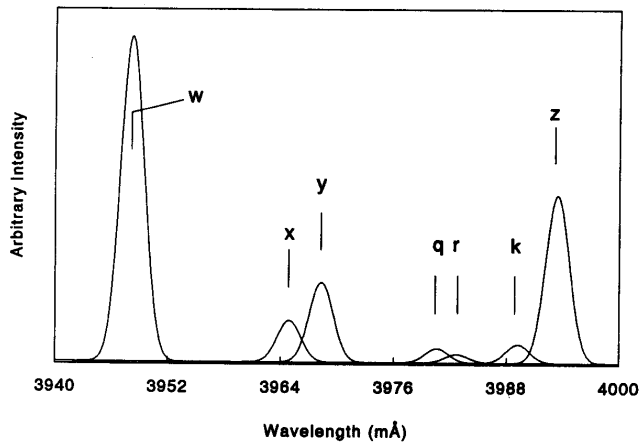

Fig. 4. The helium-like spectrum of argon during 20 msec of a single Alcator C discharge. The resonance line (w), intercombination (x and y), and forbidden (z) lines are indicated together with dielectronic (k and r) and inner-shell excitation (q) satellites.

ground-state abundance of the charge species. For a plasma in coronal equilibrium these ratios are calculated functions of T_e/T_m where T_e is the electron temperature and T_m is the temperature of maximum resonance line emission. Excitation rates have been calculated by Pradhan et al,[14] and by Pradhan and Shull,[15] which together with available data for other processes have been used to derive relative intensities for the w, x, y, and z lines for different plasma conditions. Certain line ratio combinations of the above are sensitive to the plasma parameters of temperature and density, and have been discussed in the literature.[16,17] In order to express the temperature and density dependencies in the spectrum, it is customary to define line intensity combinations; for the electron temperature

$$G = (x + y + z)/w,$$

and for the electron number density

$$R = z/(x + y).$$

In addition, $X = x/y$ can be used as a plasma independent line ratio. Auxiliary information on plasma conditions is provided by satellite lines to the He-like resonance transition due to states with a spectator electron in the $2p$ or $2s$ orbitals. The $2p$ states are formed by dielectronic recombination, and the satellite to resonance ratio is a sensitive function of T_e in a limited range for $T_e < T_m$ but independent of the ionization balance.

For our purposes here, we address only the ratios of the w, x, y, and z lines. The ratio G is a monotonically decreasing function of T_e in the range $T_e < T_m$ due to the decrease in the recombination contribution, mainly from the triplet states and the increase in resonance line emission with temperature; hence

$$\frac{\mathrm{d}\ln G}{\mathrm{d}T_e} \approx -0.23\ \mathrm{keV}^{-1}.$$

The ratio R is most sensitive to the electron density because of the collisional $2^3S_1 \rightarrow 2^3P_1$ transfer, the rate of which is proportional to n_e; hence

$$\frac{\mathrm{d}\ln R}{\mathrm{d}n_e} \approx 0.13 \times 10^{14}\ \mathrm{cm}^3.$$

The plasma spectroscopy neural network was trained with a database of 72 sets of line ratios generated from the theory, as outlined above. The data spanned electron number densities from 1×10^{13} to $1 \times 10^{15}\ \mathrm{cm}^{-3}$, and electron temperatures from 0.6 to 2.0 keV.

The experimental line ratios, determined by fitting the relevant transitions with gaussian line profiles are $R = 1.45$ and $G = 0.76$. The "best fit" to the data as determined from the neural network are $n_e = 2.3 \times 10^{14}\ \mathrm{cm}^{-3}$ and $T_e = 1.7$ keV. The accepted values for the experiment, based on other plasma measurements, are $n_e = 1.7 \times 10^{14}\ \mathrm{cm}^{-3}$ and $T_e = 1.6$ keV.[17] Researchers have previously noted large discrepancies between the plasma parameters measured by spectroscopic means and from other instruments. One possible explanation is that in the spectroscopic data analysis there is a j-line admixture to the z-line that is not accounted for; this would likely decrease the R ratio by about $\pm 15\%$.[17] They have noted also other variations that do not correlate with the measured electron density. Variations of about $\pm 20\%$ are also seen in the value of G. The most likely explanation is variations in the intensities of the intercombination lines x and y. These two lines belong to the $J = 2$ and $J = 1$ fine-structure components of the 2^3P state, which decay to the 1^1S ground state by magnetic quadruple and spin-forbidden electric dipole transitions, respectively. In the coronal and Boltzmann limits, the ratio of the x/y intensities are plasma condition independent. However, proton impact excitation may preferentially excite the 2^3P_1 state over the 2^3P_2 state; see Ref. 17 for a more detailed discussion.

CONCLUSION

In the context discussed above we are using the neural network as a multi-dimensional interpolation algorithm. Problem areas where it is hoped that the neural network technique will

be advantageous involve noisy data and partial data. The generalizing capability of neural networks should manifest itself when dealing with these issues.

A further advantage of using neural networks on a problem such as this one is that once the network is trained, however long that may take, it can compute an output for a new input in milliseconds. In addition, all the information resides in the matrix of weights so that, even if it takes many hours or days to train the network on a supercomputer, the trained network can be run on any computer.

The neural network configuration discussed above is known as a feedforward, backpropagation network. It is feedforward because all the operations consist of layer $n + 1$ operating on the output of layer n and backpropagation because the training algorithm adjusts the weights from the output layer working back toward the input layer. We have seen that such networks, when properly trained, are capable of many dimensional interpolation and, beyond this, generalization. They are capable of dealing with noisy or even missing data. Clearly the basic numerical algorithm for the training and operation of the network is quite simple. The difficult part of applying neural network techniques to a problem lies in casting the problem in an appropriate form, choosing the right variables as input and as output, and making the right choice of training data.

The conventional wisdom has been that neural networks are useful for only very rough solutions and not for accurate scientific calculations but some authors, such as Lapedes and Farber[18] refute that point of view. Indeed, the claim has been made[19] that neural networks are "formally capable of solving any conventional computational problem." There are limitations, however, due to computer hardware. Our brains have of the order of 10^{12} neurons and 10^{13} connections whereas we are limited to thousands of neurons and tens of thousands connections on general purpose digital computers. There are some gains that can be achieved using specially designed neural computers[9] and, perhaps in the future, optical neural computers.[20]

Acknowledgements—This work was performed under the auspices of the U.S. Department of Energy by LLNL under Contract No. W-7405-Eng-48.

REFERENCES

1. H. Griem, *Plasma Spectroscopy*, McGraw–Hill, New York, NY (1964).
2. J. Cooper, *Rep. Prog. Phys.* **29,** 35 (1966).
3. R. W. Lee, B. L. Whitten, and R. E. Strout II, *JQSRT* **32,** 91 (1984).
4. B. K. F. Young, W. H. Goldstein, A. L. Osterheld, R. E. Stewart, G. Charatis, and Gar. E. Busch, *J. Phys. B* **22,** L533 (1989).
5. V. A. Boiko, I. Yu Skobelev, and A. Ya Faenov, *Sov. J. Plasma Phys.* **10,** 82 (1984).
6. W. H. Goldstein, R. S. Walling, J. Bailey, M. H. Chen, R. Fortner, M. Klapisch, T. Phillips, and R. E. Stewart, *Phys. Rev. Lett.* **58,** 2300 (1987).
7. W. H. Goldstein, B. L. Whitten, A. U. Hazi, and M. H. Chen, *Phys. Rev. A* **36,** 3607 (1987).
8. J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, CA (1991).
9. R. Hecht-Nielsen, *Neurocomputing*, Addison-Wesley, Reading, MA (1990).
10. P. D. Wasserman, *Neural Computing*, Von Nostrand Reinhold, New York, NY (1989).
11. J. T. Larsen, W. L. Morgan, and W. H. Golstein, *Rev. Sci. Instrum.* **63,** 4775 (1992).
12. B. K. F. Young, R. E. Stewart, C. J. Cerjan, G. Charatis, and Gar. E. Busch, *Phys. Rev. Lett.* **61,** 2851 (1988).
13. E. Kallne et al, *Nucl. Instrum. Meth. B* **9,** 698 (1985).
14. A. K. Pradhan, D. W. Norcross, and D. G. Hummer, *Astrophys. J.* **246,** 1031 (1981).
15. A. K. Pradhan and J. M. Shull, *Astrophys. J.* **249,** 821 (1981); A. K. Pradhan, *ibid.* **263,** 821 (1982).
16. A. H. Gabriel, *Mon. Not. R. Astr. Soc.* **160,** 99 (1972).
17. E. Kallne, J. Kallne, and A. K. Pradhan, *Phys. Rev. A* **27,** 1476 (1983); *ibid.* **28,** 467 (1983).
18. A. Lapedes and R. Farber, "How neural nets work," in *Neural Information Processing Systems*, D. Z. Anderson ed., Am. Inst. of Physics, New York, NY (1988).
19. Y. S. Abu-Mostafa, "Neural networks for computing?," in *Neural Networks for Computing*, J. S. Denker ed., Am. Inst. of Physics, New York, NY (1986).
20. B. Kosko, *Neural Networks for Signal Processing*, Prentice Hall, Englewood Cliffs, NJ (1992).