24 Applied Intelligence Processing

Peter H. Sydenham
University of South Australia

Rodney Pratt
University of South Australia

24.1 Introduction

For a variety of reasons, the signal that comes from the output of a sensor will usually need to be processed, as has been explained elsewhere. This may be needed for a single sensor alone, as found in the conversion from an electronic resistance temperature sensor into the equivalent digital display number.

Alternatively and increasingly so, as is now discussed, the outputs from several sensors need to be combined into a single signal to form a mapping of those sources from "many to one." An example of this would be when a set of sensors measuring many different variables is used to determine if a machine tool is correctly making a component.

There are now many ways that can be used to combine signals, each having its own features that make it the best to use in a given circumstance. Signal-processing methods range from the well proven and mostly used method using processing based on mathematical relationships that are very precise — these are explained elsewhere. These are generally executed using digital computation and are often referred to as digital signal processing (DSP) methods. Alternatively, it has now been shown conclusively that less quantitative methods can be also used to great effect despite their lack of complete formal mathematical formulation. These are here called the (applied intelligence) methods, a convention developed to distinguish man-made systems from the very broad, and oversold, use of the term *artificial intelligence.*

Many seemingly different methods exist in the latter group but, as will be shown here, they are all part of a continuum of ways that range from use of subjective to exactly objective procedures. These are not well explained as a group because they are presented in the literature as different methods used in isolation of each other. This account shows how they all tie together, thus making it easier to decide which is appropriate in a given application. They are particularly useful for handling highly nonlinear situations where algorithms cannot be realized.

While we appear to prefer the total objectivity of a mathematically formulated method of signal processing, it is now well proven that the AI methods often are better choices to use in terms of better speed of performance and often lower cost of processing. Often they are the only solution since the algorithmic approach cannot be deployed because of the lack of a suitable mathematical formulation or powerful enough processor to run the algorithm.

Signal processing in the modern instrument, therefore, will often make use of many different methods. This account is an introduction to the characteristics of the various forms and is written to assist selection. Space limitations prevent presentation of each kind in detail.

24.2 Overview of Algorithmic Methods

Traditionally the most popular method used to develop mapping models is that of mathematical modeling. The mathematical model is usually what is sought, as it provides the highest level of understanding about the subject and the most precise representation of the behavior. The major disadvantage of mathematical models is that they can quickly become so complex that implementation of these models in measurement systems is often impractical.

In this class, the single, or set of multiple, input signal(s) to the data processor is converted to the output form using tightly formulated mathematical description. This relationship is called the algorithm. Strict relationships hold; the relationship is said to be *formal*, meaning that for any given input the output will always be the same. The algorithm supports only one interpretation.

This method of signal processing is the most highly developed method and is certainly one to aim for because it is devoid of ambiguity. All will agree on how it will respond. It carries a comforting level of understanding and, thus, acceptance.

Algorithmic methods can be very accurate, traceable, and can be calibrated with relative ease and agreement. They are the basis of many instrumentation systems. The origin of their use in instrumentation goes back to the early days of computing using, first, mechanical computational machines (late 1800s to around 1930) and then analog electric devices (early 1900s to 1960s), all of which were mostly replaced by the use of digital computers commencing around 1950. All of these algorithmic methods of processing can be simplistically regarded as embodiments of a mathematical equation inside a suitable technological machine.

As the demanded complexity and performance requirements grew over time, so did the demands on the detail of the algorithm and the means to model it inside a computational machine. Mathematical description eventually reaches limits of definition as the models push the boundaries of mathematical methods and human development. Too often, this arises before adequate detail is able to be built into the model. The algorithm is then an inadequate model of the need.

As the algorithm increases in complexity, the processing power needed must be increased to maintain both fidelity and speed of processing. Despite great advances being made in algorithm development and in computer power, the algorithmic methodology eventually encountered mathematical and technological barriers in many fields. The method is seen to not always be the best to use because of lack of an adequate algorithm or the high cost of computing.

In instrumentation, another factor also arises. Fast, detailed processing brings with it the need for increasing electrical bandwidth requirements in signal transmission. This increases implementation costs and also eventually reaches technological constraints.

24.3 Overview of Applied Intelligence Methods

Fortunately, the solutions that may overcome these limiting constraints in many circumstances were developing in other fields under the general name of artificial intelligence (now called applied intelligence in engineering), as new forms of mathematics and in other fields, such as decision theory.

Principally, a key limitation of the algorithmic method is that its unforgiving level of formalism carries with it a depth of processing exactitude that is often not warranted.

Other methods have emerged that allow vaguely subjective, as opposed to tightly objective, processing to be applied to good effect.

These AI methods have gradually gained acceptance to the degree that many are now routinely used and are supported by dedicated applications software and electronic integrated circuitry.

At first, these many alternatives were seen to be isolated methods. Gradually, the literature has shown trends to merge them in pairs. Their use in a more widely mixed form is still limited. This account seeks to give a comprehensive appreciation of the commonly met AI processing methods by placing them into relative perspective.

It is interesting to contemplate that natural world computing in animals does not appear to make much use of algorithmic methods, but does make extensive use of the methods presented here in the AI class.

The paradigm invoked here is that experience has shown that informal methods based on knowledge-based systems (KBS) can produce mappings of many inputs to one by use of less than completely formal description.

The AI methods can yield surprisingly efficient solutions to previously unsolved needs. They often can outperform algorithmic methods or carry out a similar task with far less computing power. They are all associated with multiple input processing and can be applied to forming decisions from data supplied by sensors. Each situation has to be judged on the balance between use of computing effort and effective processing.

On the downside, they lack formality and thus may be very hard to calibrate and authenticate. They, not having adequate scientific foundation and a solid formal base of operation, are not easily accepted as "sound." They are often hard to comprehend by a second party, for their description is not always adequately documented or done to any agreed convention. As their principles vary widely, they must be well understood before application is developed.

For all of these negative factors, they often are able to provide "more performance for less cost" and thus will be increasingly adopted.

Their rising level of use should not suggest the algorithmic methods will become obsolete, but more that the instrument designer now has a much larger set of processing tools available.

24.4 Mapping, in General

The high-level purpose of most signal processing is to yield knowledge of a situation so that decisions can be made.

For example, consider a health-monitoring system installed on an aircraft engine. A set of sensors of different measurand types and locations is installed at various critical points on the engine — temperatures, pressures, flow rates, metal content of the lubricating oil, and more. The data from these are collected and transmitted to a central processor using a common digital bus. The many inputs then need to be combined in some way to decide such conditions as emergency states, when to change the oil, and engine efficiency. This combination is a "mapping of many to a few."

These are not always simple mappings, for there is no adequate algorithm available to give a mathematical description for such things as degradation of oil condition. However, human intuition can be used quite effectively to obtain answers — the human mind is very capable of carrying out such mapping functions. This form of natural processing makes use of what are technically called "heuristics" — but more commonly known as "rules of thumb."

Consider the question, "How could we decide, using an automated measurement system, when loaves being made in a bakery are satisfactory to sell?" As the way to decide this almost all people asked would suggest that the weight, size, crustiness, appearance, and softness inside would be the parameters that must all be satisfactory (that is, lie within a small range of values for each) to be declared suitable. Weight and size are easily measured; the others are not for they are really heuristics, as is the choice of the set of parameters.

The thought process implemented here is that the designer starts with a desire to know something about a situation. Consider how we could automatically monitor the "risk of working in a hazardous place" in order to give an alarm at set levels. In this kind of situation a study of the problem will lead to identification of key parameters. These parameters can each be assigned safety functions that express how each parameter varies with system changes. With this framework it is then possible to set up a signal-processing system that continuously calculates the risk level. This form of solution is based on ideas embodied in the wide field of decision-making theory.

The heart of application of AI methods of signal processing in instrumentation lies with appreciation of decision-theory methods.

A range of multivariable mappings methods using AI ideas have emerged. Those well established in instrumentation are:

- Representational measurement theory and ways sets are mapped into other sets (whose usefulness is still emerging).
- Rule and frame representation of heuristic knowledge and ways they are used to form expert systems and other KBSs.
- Binary Boolean trees as crisp logical mappings; which is the foundation of the fuzzy logic method based on fuzzy set theory.

Another class of AI methodology that does not fit the same sequence, yet includes powerful methods, is one that assists optimization of the mapping setup. There are two main methods in use:

- Genetic algorithm and its use to optimize fuzzy logic and other multisensor setups, and the
- Artificial neural net, a mapping method that learns by itself, from experience, how to achieve an optimal mapping in a given situation that it has been taught to work in.

24.5 Basics of Decision Theory

Rules about a Decision-Making Process

Before entering into the detail of the AI signal-processing methods, it is necessary to develop a foundation about the ways in which decisions can be made by computers using sensed information. It is not that well appreciated, but setting up a mapping-type signal-processing situation is actually implementing a decision-making information system. General appreciation of decision making can be found in the introductory work of Kaufmann [3] .

Unlike the human brain decision maker which carries out smart thinking with ease to build a machine counterpart, an engineered object needs effective externalization of the process involved. This begins by developing appreciation of the basic rules that always apply about a decision-making situation. These have been summarized by Baker et al. [1] and condense to:

1. There must be a clearly expressed criterion for making a judgment of the options available, which must be such that others will understand how the judgment was made.
2. All decisions will involve choosing alternative strategies to arrive at the best one to use. This will involve assigning score numbers to the selected parameters and deciding how to process the set of numbers.
3. A decision is made by a choice of competing alternatives in the face of given restraints. Decisions can only rarely be made in the general sense but will be made for a given set of circumstances. The complexity of a problem rises rapidly as the number of parameters rises.
4. The process used attempts to achieve some payoff as a value added or lost. It aims to rank the various mapping alternatives to advise the apparently best to use. Note that once a decision-making mapping is built, the information about alternatives is no longer available as it will only usually embed one set of parameters as a single process.
5. A decision matrix carrying the competing parameters results. A method of combining the matrix constituents is needed, and, again, there is no singularly definitive, absolutely correct way to process the matrix.

In setting up a signal-processing mapping, these rules will need to be addressed. They will be embedded in the software of the hardware processor as its operational strategy. Considerable creativity is needed by the designer of the processor, for much of the setup of decision-making methods requires subjective human interpretation in several steps of the process. Decision making is really only needed when there is no exact and obvious answer. The devices built to mimic the human process will never be perfect. There will be much debate about which are the best methods and parameters to use. Engineers must live with this situation and make machines that will make good decisions, that are as close to perfect as possible.

Extracting Parameters

The first step in setting up a decision mapping is to understand the need. That means researching it by observation and from literature on the topic. This sets up the specific knowledge base to allow one to progress to the next step.

Then comes the need to define the key parameters of the situation. There is no organized way to develop these. They arise from inventive and innovative thought processes that seem to be based on prior learning.

To streamline this intuitive step, it is useful to apply some ordered processes that assist in externalizing appropriate parameters. Three methods are now briefly described.

Slip Writing

A group of people familiar with the problem area are read a brief statement of the problem by a person independent from the problem. An example could be "What do you think are the main parameters that a person uses to decide if a loaf of bread is fresh?"

Without much time to reflect the group is then asked to write down the key parameters that come to mind immediately as they work without talking about the problem as a group. They write down each parameter on a separate piece of paper, doing this as fast as ideas come to them. This only happens for a few minutes. The slips of paper are then collected and classified. The whole process takes around 10 min and is known as slip writing.

It will usually be found that there is common agreement about the majority of parameters with some quite unexpected ones also arising.

Slip writing is a good way to find consensus. It probes the mind well and can bring out appreciation of factors that open discussion might easily inhibit. It is important in this method to decouple the thoughts of each person during the process; otherwise the real parameters may not be externalized because some people may exert influence on others of the group.

Brainstorming and Think Tanks

If participants are shown what others are thinking and are encouraged to debate issues, it is possible to gain consensus and also allow group participants the opportunity to help each other be innovative at the same time. This method works best when the participants are prepared to go into open discussion. Several similar processes are those known as brainstorming or carrying out a think-tank session.

Here a problem in need of solution is written down as a well-prepared statement by the session organizer. A team of experts, each covering the expected aspects of the problem area, are selected and sent the statement along with any supporting exhibits. Each person considers, over a few days, how he or she might contribute a solution.

The group is then assembled. The problem is first reviewed by the session leader and each person is then asked for ideas. As ideas are externalized they are recorded in very brief form — large sheets of butcher paper are suitable. These sheets must be readable by all in the group and be prepared instantly to keep up with the thoughts of the group.

It will be found that innovative ideas will arise as candidate solutions are put up and seen by others in the group. This method encourages group-driven inventiveness.

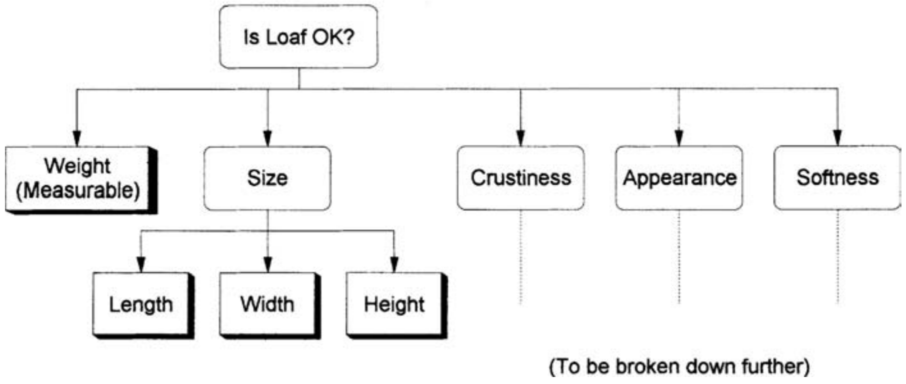


FIGURE 24.1 Knowledge trees allow facts and their relationships to be captured in pictorial form.

Gradually the group will settle on a few solutions that it feels have a good chance of succeeding. This list is then ordered in priority of likelihood of success, The session leader then writes up the outcomes, ready for further investigation.

Knowledge Trees

The final method to be described here for developing parameters of a decision, called knowledge trees, has the merit of ordering the relative place of parameters as well as encouraging inventiveness of solutions. It also provides a mapping structure. This procedure is based on the age-old realization that we think problems through by breaking them down into ever smaller subproblems until we feel able to solve them. Overall solution is then very much a matter (but not entirely so in practice) of implementing the solution of all subproblems and combining them by a process called integration.

The need is first written down. For example, “How would we measure the quality of loaves of bread?” or in a shorter form “Is the loaf OK?”

This forms the top-level parameter of the tree given as Figure 24.1. Consideration of the situation at hand then leads to realization of the collection of parameters relevant to get this answer. These might be weight, size, crustiness, appearance, and softness. They may have been externalized by a group process, such as slip writing, or created by the individual.

Each branch on the tree is then visited to see how that parameter might be measured. Only one of these can be measured as it stands — it is reasonable to assume that weight can be measured with scales.

Size is not so easy to measure as it is expressed for there is inadequate definition. More thought will yield another level to the tree — length, width, and height — for this parameter. As linear measurements, these can also be measured with ease.

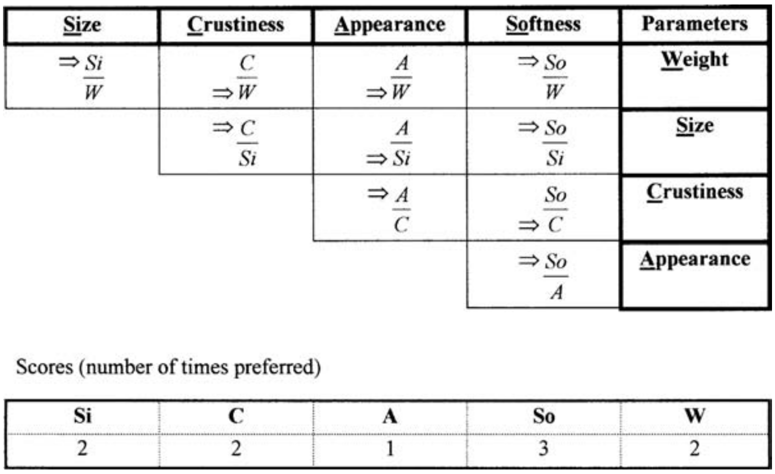
When building the branching downward, a thought process has decided how the parameters map upward and downward. Size dictates the mapping of three parameters into one, so there must also be a defined mapping model for that mapping.

Note also that to branch downward, the thought process used has actually been driven by some heuristics. Each branching has been driven by rules of some kind — but more on that in the rule-based decision-making method covered below.

It is also easy to see why the other parameters could be measured with automated instrumentation. Softness could be assessed in terms of the squeeze factor, which is actually measurable as the compliance of the loaf at the center point of a side of the loaf. Appearance would map down the tree into color, texture, and graininess of the image. It is left to the reader to think up and draw a complete tree.

When all branch ends have been reticulated down to the point where they can be measured, the system mapping can be implemented with a human-made sensing system. The parameters are externalized and the mapping process is largely decided.

Use of tree-based thinking is a simple, yet powerful, way of keeping track of decision making for a complex situation. The recorded form also allows others to see the thought process used.



Size	Crustiness	Appearance	Softness	Parameters
$\frac{\Rightarrow Si}{W}$	$\frac{C}{\Rightarrow W}$	$\frac{A}{\Rightarrow W}$	$\frac{\Rightarrow So}{W}$	**Weight**
	$\frac{\Rightarrow C}{Si}$	$\frac{A}{\Rightarrow Si}$	$\frac{\Rightarrow So}{Si}$	**Size**
		$\frac{\Rightarrow A}{C}$	$\frac{So}{\Rightarrow C}$	**Crustiness**
			$\frac{\Rightarrow So}{A}$	**Appearance**

Scores (number of times preferred)

Si	C	A	So	W
2	2	1	3	2

FIGURE 24.2 Triangle of pairs assessment is a simple way to decide which choice to make. This table gives the workings for the grading of bread as in the example of Figure 24.1

Two Examples of Decision-Assistance Methods

Triangle of Pairs

Having now seen how to externalize parameters and how they might be interrelated using trees, we can move on to investigate how to set up a suitable decision-making process.

Knowing the parameters is not enough yet to design a multisensor mapping processor. The relative importance of the parameters is also a key factor. Furthermore, the tree is not the only way to combine sensor signals.

Two, of many, examples of decision-assistance methods are now outlined to illustrate these points.

The first, the triangle of pairs (TOP) method, allows parameters to be ranked against others on the binary-only basis of which is preferred of each two compared. In the bread example, compare some of the parameters for their relative importance. Crustiness is preferred to weight. Softness is preferred to size and so on until all pairs have been considered. If all combinations are carried through and recorded as a matrix, a triangle of pairs results, as in Figure 24.2.

Having formed a matrix of competing parameters, the next step is to decide how the matrix can be processed. This is where much debate can occur. For the TOP method, however, the binary nature of the choices means a simple count of first preferences gives the ordered preference of parameters — at least as that person assessed it!

We will see later how the idea is extended by giving parameters a varying degree of "preference" rather that the simple binary choice allowed here.

Utility Analysis

A more fully developed method for making decisions in complex and subjective situations is one called utility analysis (UA). This is a process that can be applied to find the usefulness of a design, piece of equipment, or any similar situation where one can externalize and prioritize a set of measurable parameters. Although mostly applied to decision making as a paper study, the process is amenable to the creation of a multisensor mapping processor.

Appreciation of this process is easily obtained by working through an example. Consider the need to set up a method for grading the quality of bread made in an automated bakery.

The first step is to decide the parameters that will affect the choice finally made. We select weight, size, and appearance as the key parameters to illustrate the method. (More parameters might be needed in a real situation.) We also decide that these are important according to the relative weighting ratios of 1.0:0.2:0.8.

Next, utility curves must be set up, one for each parameter. These show how the usefulness of the parameter changes as the parameter ranges.

The simplest way to obtain these graphs is to use one's own intuition, but a better way is to make use of some form of consensus-forming procedure as discussed above. Figure 24.3 shows what these three functions might look like. As a guide to their construction, if the weight of the loaf is too low, it fails to comply with legal requirements and thus has zero utility as a product below the allowed uncertainty of weight. The size factor depends on the type of bread. Here it is assumed it is for sandwich making, in which case the size can be too small for sliced meats or too big for a toaster. Appearance can only be measured by mapping downward to a set of measurands; it is convenient to plot the function in more vaguely defined terms in this method.

Note that the weighting ratios have already been incorporated into the utility charts by setting their best values at the correct percentage.

It is necessary to reinforce the fact that the set of graphs is needed for the parameters, not for each case to be considered. With the charts set up, the selection process can begin.

A matrix is now created, as also shown in Figure 24.3. The actual measured weight value (960 g) for loaf 1 (under automatic inspection) is compared with the graph of weight to yield a utility of 0.4. The size is done likewise, using the size graph to get 0.2, and the appearance sensor set tells us it is between poor and good to give us 0.4. Each loaf is subjected to the same process. The other two loaves have different sets of scores.

The combined usefulness of a given loaf is now to be decided by processing the set of numbers for that loaf. Here is where some difficulty arises, because there are many ways to combine the three scores for each loaf. One way often used is simply to sum the values, as is done in the example. Note that this can be satisfactory unless a zero or other unacceptable value arises, in which case more mapping processing is needed. Assume that an acceptable combined score has been determined to be 1.8 or higher (with the best, due to the weightings, 2.0).

Having carried out the processing of the matrix, we find that the first loaf is not acceptable and the other two are equally acceptable.

What we have done is to form an automatic measuring system that can make assisted and graded decisions. There is certainly a degree of human subjectivity inherent in the method, but by recording what is taking place the task can be automated and it can also be evaluated in the light of experience, correcting choices of parameters and weightings.

Other decision-making methods exist: they use elements of the above concepts in a variety of assemblages.

The above has been a very rapid introduction to decision-making methods. These are rarely taught as the foundation of the now popularized AI signal-processing methods given next. They are, however, the conceptual basis of the AI processes.

24.6 Principal AI Methods

Measurement Theory

Sensors observe a finite number of variables associated with real-world events. Measurement systems are required to convert the information from these sensors into knowledge. This process often involves what is known as mapping.

Depending on the loss or preservation of information, the mapping process can be classified as either

1. transformation, or
2. abstraction.

If a rule exists that assigns every element of x in the set X (written $x \in X$) to precisely one element $y \in Y$, we say that a function exists that maps the set X to the set Y, and this is represented as Figure 24.4 and symbolically by

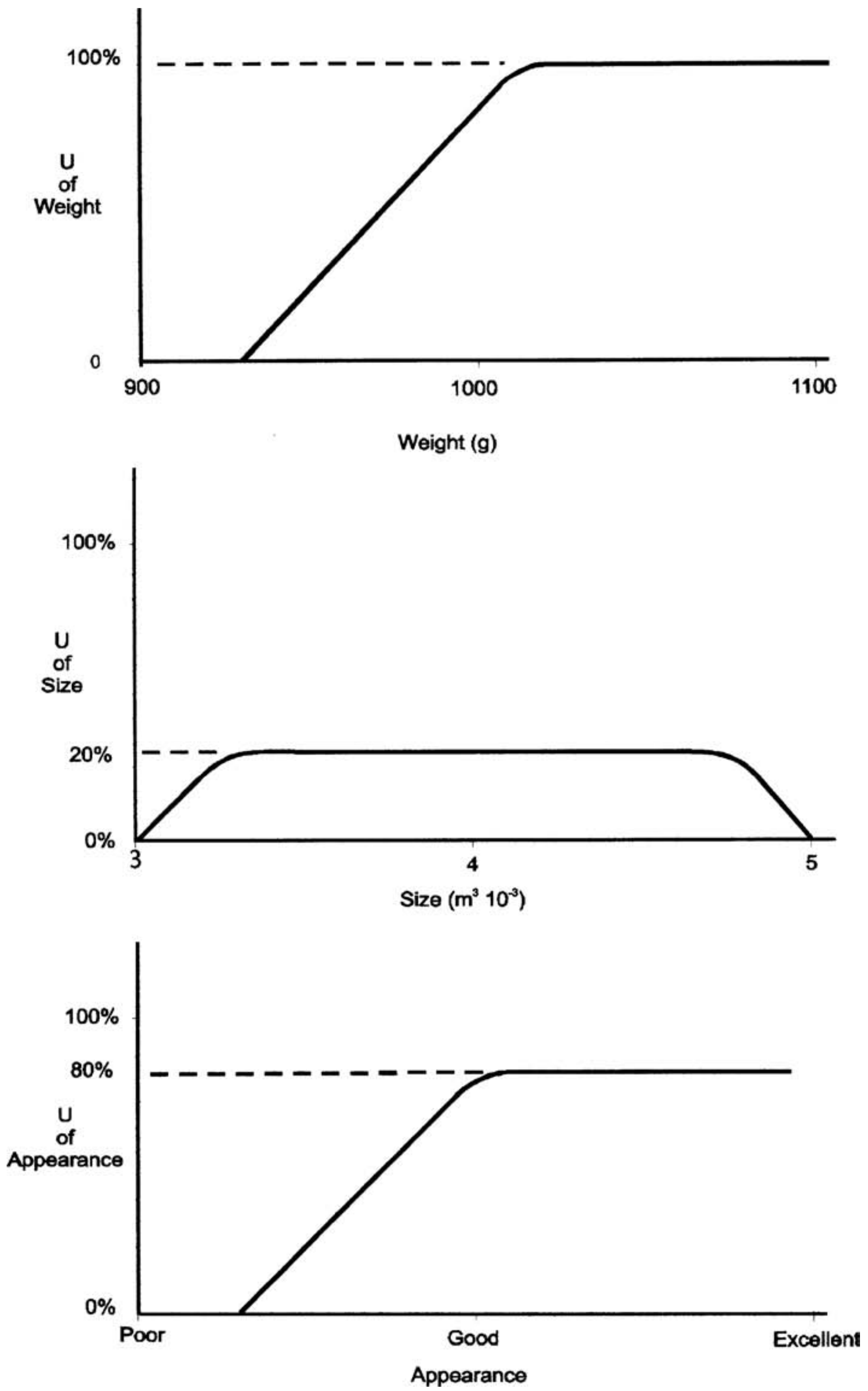


FIGURE 24.3 Utility analysis is a more detailed way to automate loaf inspection with a set of sensors. Here are shown three of the utility functions for the example in Figure 24.1 along with the scoring matrix for loaves passing the inspection point.

Parameter / Object Under Assessment	Weight (g)	Size (m^3 10^{-3})	Appearance (grade)	Score (Σ)
Loaf 1	(960) 0.4	(4) 0.2	(poor/good) 0.4	1.0
Loaf 2	(1000) 1.0	(3.5) 0.2	(good/excellent) 0.8	2.0
Loaf 3	(1100) 1.0	(4.9) 0.1	(excellent) 0.8	1.9
•	•	•	•	•
•	•	•	•	•
Loaf 'n'	etc	etc	etc	etc

FIGURE 24.3 (continued)

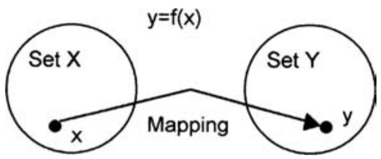


FIGURE 24.4 Pictorial representation of the set theoretical mapping process.

$$y = f(x)(x \in X)$$

A transformation defines a one-to-one mapping on the set of all elements of x into y, in which for all $x \in X$ there exists a unique value in Y; therefore the inverse transformation is guaranteed, i.e., $x = f^{-1}(y)$.

An abstraction differs from a transformation in that it can map a number of elements $x_i \in X$ into the same y, or a many-to-one mapping; hence, no inverse operator can exist such that the inverse image of X can be uniquely retrieved from Y. Therefore, abstraction is usually characterized by a loss of information.

It is possible to define the set theoretical relationship as a formal mathematical model and then embed that in the signal processor. Although this is a clearly useful method of formalizing mappings, this approach is not commonly used at the time of writing but can be expected to find more favor in the future.

Rule- and Frame-Based Systems

In early stages of problem solving, we seem naturally to look to rules of thumb to get the solution started. Even the algorithmic methods start with this form of thinking, for one has to decide what the elements of the decision are and how they might be assembled into a strategy for implementation in the signal processor. As the rules are externalized one's thought patterns also usually build knowledge trees that show the relationship between the rules.

Consideration of the knowledge supporting the structure of a tree will reveal that the decision needed about which way to branch as one moves through a tree is actually the implementation of a rule that has relevance at that junction. Rules link parameters together.

Figure 24.5 is a tree giving some of the design options for improving the performance of an audio speaker system. This tree has been built by applying a designer's knowledge of speaker system design. The heuristic rule set for the top branch is stated as

IF	speaker output (W) increases
AND	distortion (D) is reduced

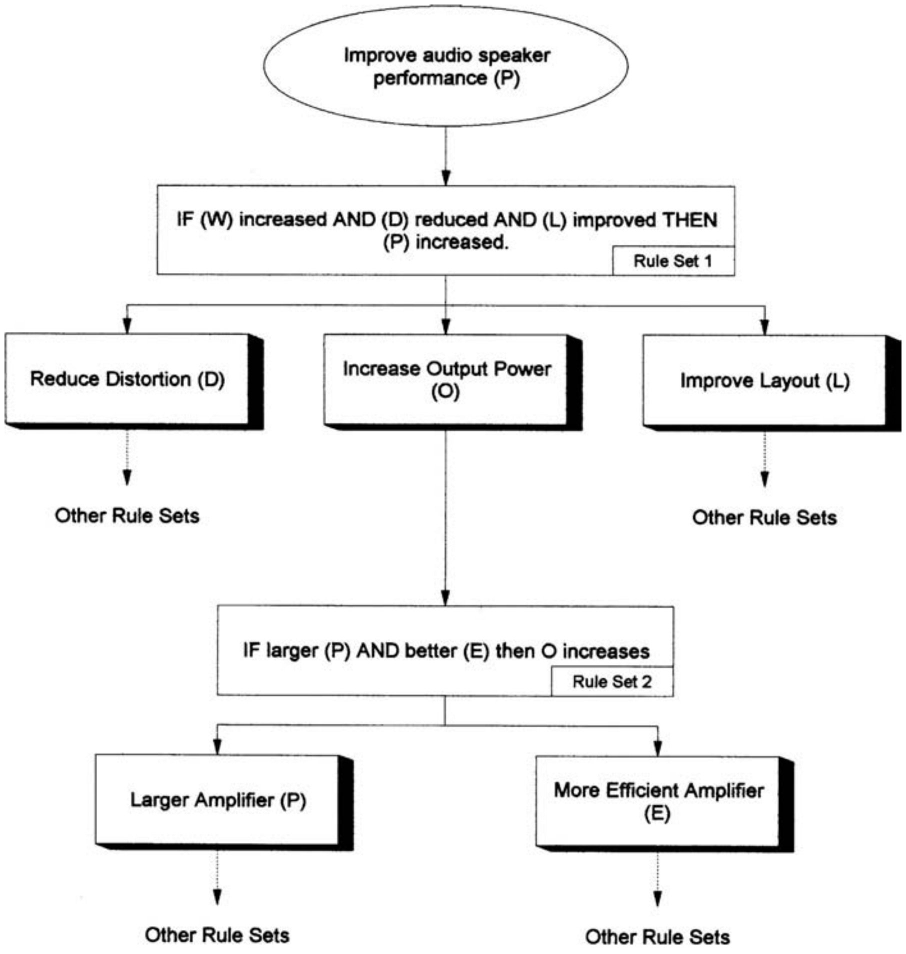

Improve audio speaker performance (P)
IF (W) increased AND (D) reduced AND (L) improved THEN (P) increased.
Rule Set 1
Reduce Distortion (D)
Increase Output Power (O)
Improve Layout (L)
Other Rule Sets
Other Rule Sets
IF larger (P) AND better (E) then O increases
Rule Set 2
Larger Amplifier (P)
More Efficient Amplifier (E)
Other Rule Sets
Other Rule Sets

FIGURE 24.5 Part of a design knowledge tree for improving the performance of an audio speaker system.

AND positioning (L) improved
THEN audio output (O) is improved

No part of the rule set could be first realized using formal mathematical or algorithmic thinking. Intuition, leap, common sense, and other terms that describe the human intelligence process must be applied to commence a solution.

At some stage a rule may become describable by an algorithm — when that occurs a formal mathematical expression can be used to embed relationships. However, this is often not the case and so methods have been developed in computers to process heuristics.

The full set of AI techniques were originally all rolled into what became known as KBSs but this term is so overused that it has lost specific meaning.

Among the first AI processing methods were special computing ways to process the logic of a set of rules. These became known as expert systems (ES). In the example above, the rule tree is very sparse; a practical system for decision making is likely to have from 100 to several thousand rules.

The rules are considered by an inference engine (a software program) that is able to carry out Boolean logical operations of AND, OR, etc. to yield the outcome appropriate to the set of rules relevant to the problem at hand.

Trees can be traversed from the top down (downward chaining) or from the bottom up (upward chaining) and modern ES software applications carry out these operations with great sophistication.

KBS methods generally suffer from the feature that they seem to give answers all too easily, for they use only a few of the many rules available to come to a solution in a given situation. To help users feel more confident in their application, features are often offered that include plotting of the chaining used to get the solution or stating the rule set used.

Rule-based software applications are sold as empty shells. The user fills the shells with the rule set to make the application specific. These applications are now commonly used. They are relatively easy to use without the need for a competent computer programmer.

Rules are a primitive way to express knowledge. A better form of representation is the frame. This has the ability to hold more knowledge than a single rule and is a small database about a limited area of the system of interest. Frames are like objects in object-oriented programming. Advanced ES shells operate with frames.

ES shells are now very capable entities for decision making. They are a significant tool in the instrument signal processor's toolbox. Space restricts more explanation but enough has been stated here to allow further development of AI methods.

Some basic characteristics about rule- and frame-based processing are as follows (these apply variously):

- The software program is often self-explanatory as the rules can be read as (almost) normal language.
- They need considerable computer power as the rule number increases.
- They are relatively slow to yield a solution but are best used for cases where slow outcomes are applicable.
- They need special software.
- They are not that well known, so their application may be slow to find favor.

ESs soon run out of usefulness if the problem becomes complex. The computer search becomes too slow because the number of rules needed rapidly rises with problem complexity. The approach used today for large-problem-solving systems is to build an ES for each facet of the problem. These small-problem AI units are called agents. A set of agents is then combined using a conceptual software-based blackboard that calls on the agents to investigate a problem put to the system.

The chaining operation basically only carries out simple Boolean algebra operations using system parameters represented by a description called a rule. The system has no understanding of the wording of the rule. Thus, it is only processing as though the tree branching is either one way or the other. In its simplest form, it contains no concept of making that branching decision with a graded concept of which way to go.

Real life is full of unclear logical operations. The outcome of a decision will be clearly this way or that, but just where the change arises is problematic. The changeover point is unclear because the variables of the rule are fuzzy. It is desirable to process rules with regard of their likelihood of relevance depending on the state and selection of other rules. As is explained below, the fuzzy logic method of carrying out a mapping is another way that allows the rule to have more variability than the two-state exactness of binary logic.

Fuzzy Logic

This explanation is not intended to be a rigorous mathematical examination of fuzzy sets and fuzzy logic but rather explain, through example, the application of fuzzy techniques in measurement systems, specifically in respect to mapping models. For more detail, see Mauris et al. [4].

The simple example used here is a measurement system, Figure 24.6, that maps two input sensors (temperature and humidity) into one output value (comfort index). Clearly there is no formally accepted definition of comfort index as it is a subjective assessment. One of the advantages of fuzzy sets is that they are usually intended to model people's cognitive states.

In the mid 1960s Professor Lofti Zadeh recognized the deficiencies of Boolean logic, in that its TRUE/FALSE nature did not deal well with the shades of gray that exist in real life situations.

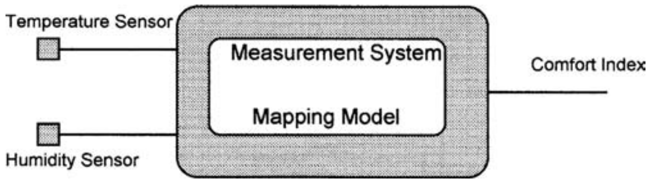


FIGURE 24.6 A simple comfort index measurement system uses temperature and humidity variables.

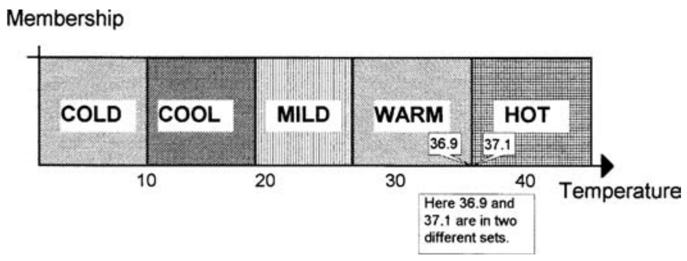


FIGURE 24.7 Conventional crisp set for the measurement system of Figure 24.6.

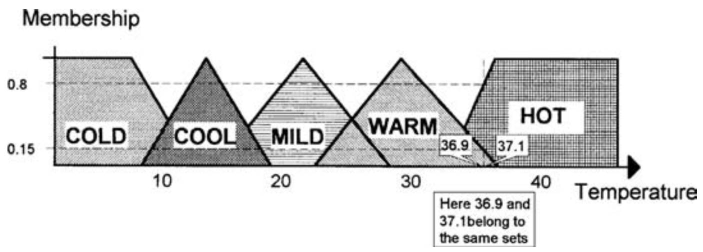


FIGURE 24.8 Fuzzy set representation temperature regimes in the comfort controller example.

Boolean logic uses classical set theory where an element is either viewed as entirely true or completely false $A = \{0,1\}$. These are often referred to as a crisp sets, Figure 24.7. In a crisp set the transition between sets is instantaneous, i.e., 36.9°C is considered warm whereas 37.1°C is considered hot. Hence, small changes in the input values can result in significant changes in the model output. Clearly, the real world is not like this.

Fuzzy logic uses a multivalued set where degrees of membership are represented by a number between 0 and 1 $A = [0,1]$ $\mu_{A:}$ $U \to [0,1]$, where μ_A is the membership function. With fuzzy logic the transition between sets is gradual and small changes in input values result in a more graceful change in the model output, Figure 24.8.

Fuzzy Expert Systems

A fuzzy expert system, Figure 24.9, combines fuzzy membership functions and rules, in place of the often all-too-crisp Boolean logic, to reason about data. The fuzzy expert system is usually composed of three processing sections:

Step 1. Fuzzification
Step 2. Rule evaluation
Step 3. Defuzzification

Step 1 — Fuzzification.
In fuzzification crisp inputs from input sensors are converted into fuzzy inputs using the membership functions in the knowledge base. A fuzzy input value is generated for each linguistic label of each input. For example, in Figure 24.8, for an input temperature of 37°C the fuzzy input is COLD(0.0), COOL(0.0),

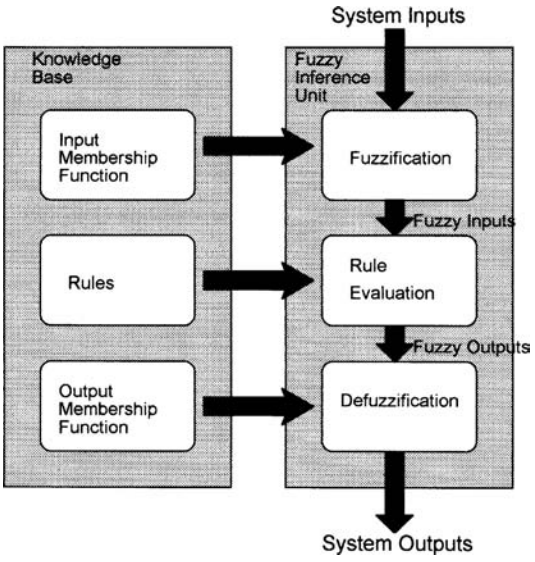


FIGURE 24.9 A fuzzy inference system combines rules in a way that allows them to be fuzzy in nature, that is, not crisp.

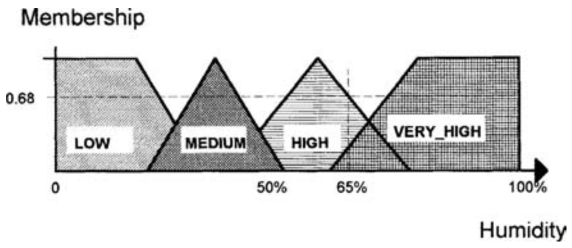


FIGURE 24.10 Humidity membership function for the controller example.

MILD(0.0), WARM(0.15), HOT(0.80). A similar set of fuzzy inputs are generated for the humidity sensor, Figure 24.10.

Step 2 — Rule Evaluation.
Rules provide a link between fuzzy inputs and fuzzy outputs. Rules are usually expressed in the form of IF … AND/OR … THEN … statements.

For example,

'IF' the TEMPERATURE is HOT 'AND' the HUMIDITY is HIGH 'THEN' it is UNCOMFORTABLE.
'IF' the TEMPERATURE is MILD 'AND' the HUMIDITY is MEDIUM 'THEN' it is COMFORTABLE.
'IF' the TEMPERATURE is WARM 'AND' the HUMIDITY is LOW 'THEN' it is UNCOMFORTABLE.

Rules can also be expressed in the form of a table or matrix, called the Fuzzy Associative Matrix. This matrix, Table 24.1, provides a complete description of the system performance for all combinations of inputs.

The function of the rule evaluation step is to evaluate the relative strengths or truth of each of the rules in order to determine which rules dominate. In this example the rules contain AND relationships and therefore the overall rule strength must be the minimum (MIN) value of the two strengths of the input values.

TABLE 24.1 The Fuzzy Associative Matrix Links Inputs and Outputs

Hum/Temp	Cold	Cool	Mild	Warm	Hot
Low	Uncomfortable	Uncomfortable	Uncomfortable	Uncomfortable	Uncomfortable
Medium	Uncomfortable	Acceptable	Comfortable	Acceptable	Uncomfortable
High	Uncomfortable	Acceptable	Acceptable	Acceptable	Uncomfortable
Very high	Uncomfortable	Uncomfortable	Uncomfortable	Uncomfortable	Uncomfortable

For example, at a temperature of 37°C and a humidity of 65% the rule strengths of the three example rules are:

'IF' the TEMPERATURE is HOT(0.8) 'AND' HUMIDITY is HIGH(0.68) 'THEN' it is UNCOMFORTABLE (Rule Strength = MIN(0.8,0.68) = 0.68).
'IF' the TEMPERATURE is MILD(0.0) 'AND' the HUMIDITY is MEDIUM(0.0) 'THEN' it is COMFORTABLE (Rule Strength = MIN(0.0,0.0) = 0.0).
'IF' the TEMPERATURE is WARM 'AND' the HUMIDITY is LOW 'THEN' it is UNCOMFORTABLE (Rule Strength = MIN(0.15,0.0) = 0.0).

All rules must be evaluated (in this case all 20 in the matrix) to determine each rule strength. If two rule strengths exist for one fuzzy output label, then the maximum (MAX) rule strength is used because this represents the rule which is most true; that is, in the previous example the fuzzy output for UNCOMFORTABLE is the MAX(0.68,0.0) = 0.68.

Step 3 — Defuzzification.
Now that rule strengths exist for all the output fuzzy labels, a crisp output value can be determined from the fuzzy output values. The most common method used to defuzzify the fuzzy output value is the center of gravity (COG) method. The fuzzy rule strengths determined from rule evaluation are used to truncate the top of the output membership functions. Given this area curve, the COG or balance point can then be calculated. For example, in Figure 24.11, for fuzzy output values of

UNCOMFORTABLE = 0.68
ACCEPTABLE = 0.2
COMFORTABLE = 0

the COG or crisp "Comfort Index" evaluates to 25.

Fuzzy logic signal processing is now a well-developed method. It is supported by copious teaching material including those on the Internet and on CD ROMs provided by manufacturers of this form of special integrated circuit chip sets and setup software. Possibly its best known application has been in clothes washing machines where the wash parameters are set to suit the load. Although there are still limitations, this method can be considered to be a mature procedure.

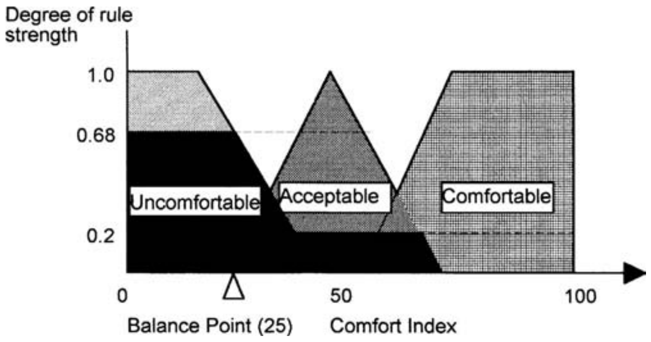


FIGURE 24.11 Output membership functions for controller. Each type of shaded area represents the three states of comfort.

Genetic Algorithms

The methods discussed thus far all have required the user to advise the system about the parameters to use. That is, they need to be taught efficient mappings.

In sharp contrast to these types of systems, there also exist other AI methods that possess the ability to learn, by themselves, what are the more optimal mapping configurations. Two main techniques are used — genetic algorithms (GAs) and artificial neural networks (ANNs). These self-learning processes both work well in certain situations and are now commonly used in signal processing. On the face of it, both seem to possess magical properties because they defy logical thought processes and a clear understanding of how they actually operate.

We begin with an overview of GAs. These make use of the basic principles by which natural selection found in living things, that is, from genetics, is able to improve gradually the fitness of species. That the genetic principles found in nature can be used in human-made applications is accredited to pioneering work of John Holland in the mid 1970s. Today, it is a large field in algorithm optimization research; see Tang et al. [7]. A very simplistic description now follows to give some insight.

The concept starts with the selection of a set of features, Figure 24.12 (these can take a wide range of forms and are not just measurement parameters), that represent the essential features of a system of interest. As examples, the DNA molecule carries the code of the characteristics of living beings and a computer string can carry a coded message that represents the features of the behavior of some human-devised system.

Various types of events (crossover, mutation, inversion are commonly encountered methods) can slightly alter the code of any particular string. When this happens, the new string then represents another closely similar, but different system having new properties. Consider, next, that a number of slightly different code strings have been formed.

When a change takes place in the code of a string, it is assessed against the other strings using rules for a predecided fitness test. If improvement has occurred in the overall properties, then it is adopted as one of the full set. If not better, then it is discarded. In this way the set of strings, and thus the total system capability, gradually improves toward an optimal state.

The operational aspects of such systems are beyond description here. Suffice to say that the technique is well established — but highly specialized — and is undergoing massive international research effort in hope of alleviating certain limitations.

The first limitation is that although it is generally agreed each adopted change for the better takes the overall system capability closer to the goal being sought, there is, as yet, no theory that can be applied to show the state of maximum optimization. GAs, therefore, always have doubt associated with their solutions as to how much more improvement might be possible.

The second limitation becomes obvious when the computational demands are considered in terms of the number of comparison operations needed to be run in the improvement process. This number can be truly huge, especially as the range of options rises with increase in string length. This kind of operation usually needs very large and fast computing power. In cases where there is plenty of time to determine an improvement, this is not a major limitation. An example of effective use would be finding how best to operate a wide range of functions in a complex system when the task is then fixed; this method was used to set up more optimal use of power supplies in a space vehicle. In cases where the variables in a code string are fast changing, the method may not be applicable.

When running this form of computer program, it is essential to have a measure of the speed — the dynamic behavior — at which this highly iterative process is moving toward the optimization goal: the user can then decide if it is a viable method for obtaining improvement.

With this background, it is probably obvious why GAs are sometimes used to set up the membership functions of fuzzy logic systems. As explained above, the membership functions each are part of a set of individual functions that form the mapping for a multisensor system. The choice of the membership function is largely heuristic and thus may not be the best function to use. By setting up several sets of functions it is then possible to apply GA computing methods iteratively to select a better set to use. Such

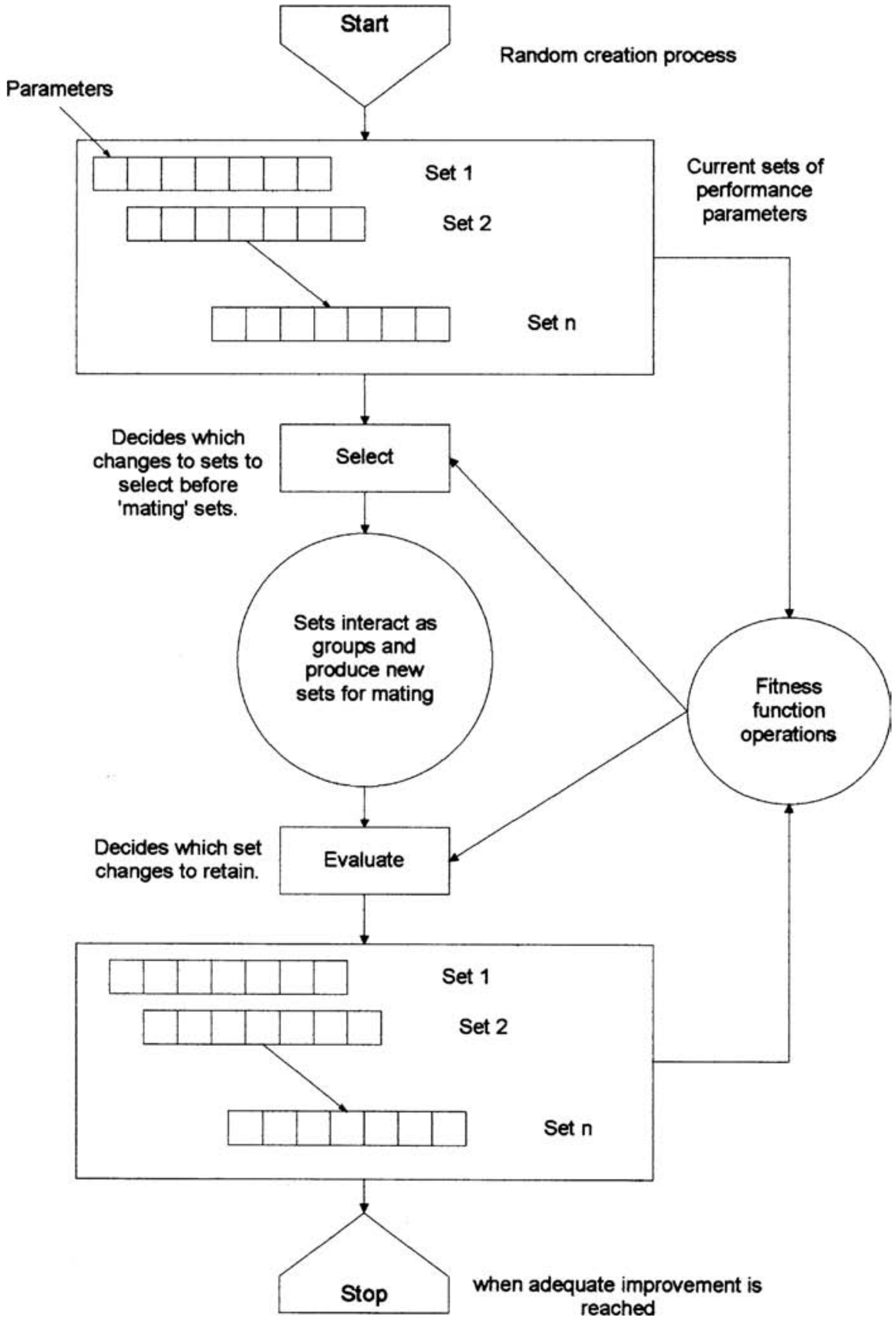

Start
Random creation process
Parameters
Set 1
Set 2
Set n
Current sets of performance parameters
Decides which changes to sets to select before 'mating' sets.
Select
Sets interact as groups and produce new sets for mating
Fitness function operations
Decides which set changes to retain.
Evaluate
Set 1
Set 2
Set n
Stop
when adequate improvement is reached

FIGURE 24.12 In the GA method sets of code strings are first modified by some form of genetic operations. They are then intercompared using fitness functions to select a better code to use in the subsequent set of strings. This iterative process is continued until some event disrupts it.

systems have been used where the time of currency of a function set is longer than the time needed to improve the functions. Obviously, use of GAs increases system set up and operational complexity, but once implemented may yield worthwhile gains.

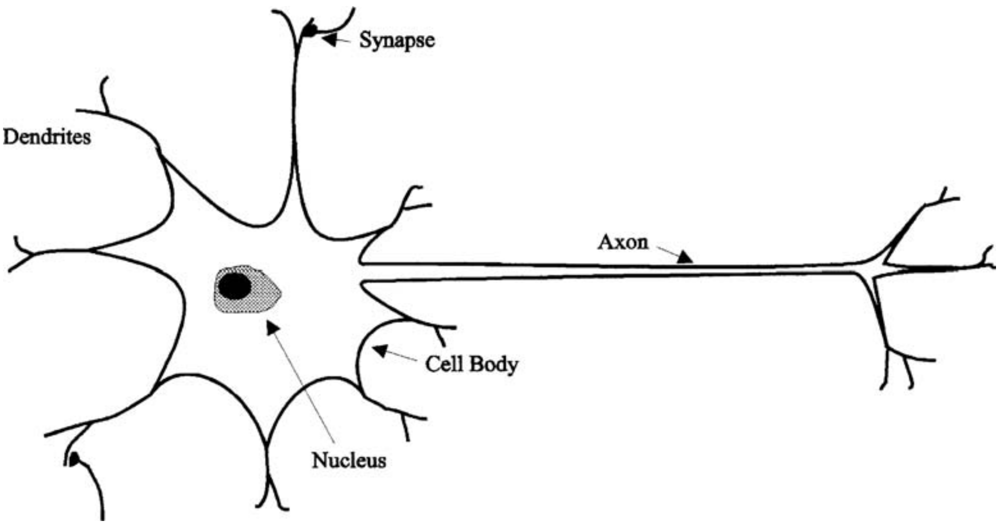


FIGURE 24.13 A biological neuron.

Artificial Neural Networks

The AI basket of potentially useful methods in signal processing is full of surprises. Attention is now directed to another method, which can learn, after an initial training period, how to better operate a given mapping process.

Neurocomputing or use of ANNs is another AI technique well suited to make a mapping processor in some circumstances. (These are sometimes called NNs but should not be used for human-devised systems as that leads to confusion with life sciences research on living neural networks.) ANNs are particularly useful when mapping complex multidimensional information to a simpler representation. Because of their ability to deal with nonlinear relationships, they find application in areas where traditional statistical techniques are of limited use. Some application areas include pattern recognition and classification, categorization, function approximation, and control.

Biological Neuron

The ANN has been inspired by the biological structure of the brain, and it is an attempt to mimic processes within the biological nervous system. The neuron is an information-processing cell in the brain, Figure 24.13. It consists of:

1. A body or *soma*
2. Input branches or *dendrites*
3. Output branch or *axon*

The neuron receives signals through its dendrites, and then transmits signals from its cell body along the axon. The neuron will generate an output (or fire) when the aggregation of the inputs reaches a threshold level. At the terminals of the axon are *synapses.* The synapse couples the signals from the axon of one neuron to the dendrites of another neuron. The strength with which these signals are transferred from the axon to the dendrite is controlled by the synapse and can be altered; hence the synapses can learn from experience.

Desirable characteristics of neural systems include:

- Massive parallelism
- Learning ability
- Ability to generalize

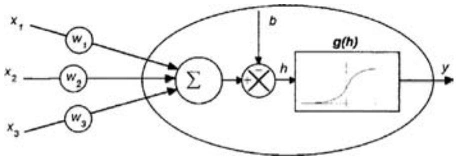


FIGURE 24.14 This neuron model is commonly used.

- Adaptability
- Fault tolerance

It is the attempt to construct machines that exhibit these characteristics that has led to the ANN methods of signal processing.

Artificial Neural Network

In 1943 McCulloch and Pitts proposed the first model of an artificial neuron. This has formed the basis for the generally accepted form of synthetic neuron model or processing element (PE); see Figure 24.14.

The output of the processing element y is given by

$$y = g\left[\left(\sum_i w_i \cdot x_i\right) - b\right]$$

where x_i are the PE inputs with weights (synaptic strengths) w_i, b the PE bias, and g the activation or transfer function. Many types of function have been proposed but the most popular is the sigmoid function, defined by

$$g(h) = \frac{1}{\left(1 + e^{(-\beta h)}\right)}$$

where β is the slope parameter.

By themselves the processing elements are very simple; however, when the individual elements are joined into large interconnected networks, complex relationships can be represented. Although a number of network architectures [2] exist, one of the most commonly discussed architectures found in the literature is the feed-forward multilayered perceptron. Figure 24.15 provides a simple illustration of how

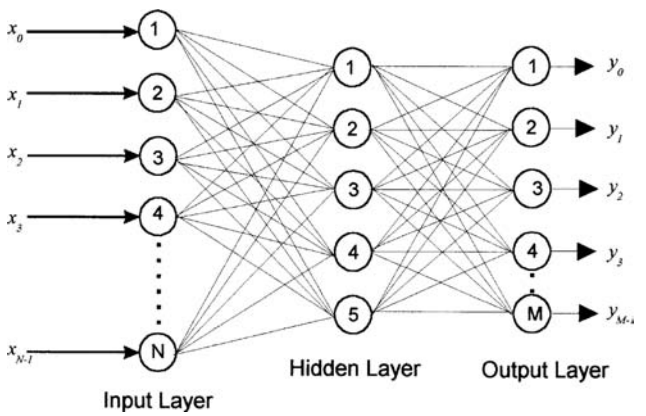


FIGURE 24.15 A typical three-layer neural network as is commonly used to create an optimal mapping from sensors to outputs.

a multilayered ANN maps a multidimensional input vector $x_0, \ldots x_{N-1}$ in an input space to a vector $y_0, \ldots y_{M-1}$ in an output space.

Learning

A fundamental characteristic of the ANN, once it has been set up as an operational tool in software form, is that it does not need to be programmed for the application. ANNs appear to learn rules from a representative set of examples, rather than having rules programmed in by an expert. The knowledge acquired by the system that controls how the system maps input to output is held within the connection weights of the network.

The focus of extensive ongoing research is the search for optimal training techniques. These techniques tend to fall into two broad categories — supervised and unsupervised learning.

In supervised learning, a representative set of inputs is presented to the network which then modifies its internal weights in order to achieve a desired output. With unsupervised learning, the input data only are presented to the network, following which the network organizes itself through self-modification of its internal weights so that it responds differently to each input stimulus.

It is beyond the scope of this text to review all the current learning techniques for ANNs. This is well documented in the literature [8]. One popular training algorithm — backpropagation — will be discussed as an example.

Backpropagation

Backpropagation is an example of a supervised learning paradigm commonly used with multilayer perceptron network architectures. Backpropagation follows the error-correction principle and uses the error signal {d (desired output) – y (actual output)} to modify the connection weights to reduce this error. The backpropagation algorithm is implemented as follows:

1. Initialize network weights to small random values.
2. Present an input vector $x_0, \ldots x_{N-1}$ and the corresponding desired output vector $d_0, \ldots d_{M-1}$.
3. Calculate the actual output vector $y_0, \ldots y_{M-1}$ by propagating the input through the network.
4. Use a recursive algorithm starting at the output layer and adjust the weights backward by

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x_i'$$

where $w_{ij}(t)$ = the weight from an input to node j at time t
x_i' = either the output of node i or an input
η = a gain term ($0.0 < \eta < 1.0$)
δ_j = an error term for node j

For the output layer $l = L$ the error is calculated:

$$\delta_j^L = g'(h_j^L)[d_j - y_j]$$

where h_j^L = the net input to the jth unit in the L layer
g' = the derivative of the activation function g

For hidden layers $l = (L - 1), \ldots, 1$. the error is calculated:

$$\delta_j^L = g'(h_j^l) \sum_l w_{ij}^{l+1} \delta_j^{l+1}$$

where h_j^l = the net input to the jth unit in the lth layer.
g' = the derivative of the activation function g

5. Return to Step 2 and repeat for the next pattern until the error reaches a predefined minimum level.

Unfortunately, no method exists that allows the ANN to create or learn information that is not contained in the training data; that is, the ANN can only reproduce based on experience. Under certain conditions, however, the network can generalize, that is, approximate, output values for data not contained in the training set.

The neural network can be considered as a universal approximator. During supervised learning, the output eventually approximates a target value based on training data. While this is a useful function, the ability to provide output data for test cases not in the training data is more desirable. Loosely, generalization can be viewed in terms of interpolation and extrapolation based on training data. If a test case is closely surrounded by training data, then (as with interpolation) the output accuracy is generally reliable. If, however, the test case is outside of, and not sufficiently close to, training data then (as with extrapolation) the accuracy is notoriously unreliable. Therefore, if the training cases are a sufficiently large sample of the total population of possible input data so that each test case is close to a training case, then the network will adequately generalize.

While multilayer feed-forward networks are finding increasing application in a wide range of products, many design issues such as determining an optimal number of layers, units, and training set for good generalization are research topics. Current theory provides loose guidelines and many of these design issues are resolved by trial and error. Another disadvantage of ANNs is the high demand that many training algorithms can put on computing resources because of their recursive nature.

The ANN, then, provides another alternative for development of suitable mapping models for measurement systems. They can be used to describe complex nonlinear relationships using a network of very simple processing elements. The attraction of the ANN lies in its ability to learn. As long as there exists a sufficiently representative sample of input-to-output data available, the mathematical relationship of the mapping function need not be known. It is effectively taught to the network during a learning process.

Again, there exist important limitations. The worst is the time it might take to adjust the system nodes and weights to a nearly final state. This can often require considerably more time than the time-varying properties of the inputs allow. In many potential applications, they take too long to learn and are not effective. Again computational speed and power are governing factors.

Despite their shortcomings in some applications, ANNs are now a commonly used procedure to set up sensor mapping systems. Examples are banknote image detection and the increased sensitivity of the detection of aluminum in water.

24.7 Problems in Calibration of AI Processing Methods

Calibration of a measurement system is the result of using an agreed upon, often legally binding, process by which it is proven to possess a declared level of accuracy in its measurement outcome. In conventional instrument terms this implies the system can be set up and compared with a measurement method of superior performance to give its error of accuracy plus its variance from the agreed value determined with a level of uncertainty. This kind of instrument system is then accepted to have a known behavior that could be explained by the laws of physics as causal and unique in performance. The prime example, the physical standard apparatus for a parameter, can be and is defined such that it will always give very closely the same outcome, even if built in different laboratories. It will have predictable behavior. At the heart of this acceptability is that it can be modeled in terms of an algorithm. All parts in it follow formal laws and have the same outcomes from implementation to implementation.

Most of this notion has to be put aside because, as explained above, AI-based instrument signal processors are built on a raft of transformations that convert subjective situations into objective ones or they carry out unexplained processes. It is, therefore, not hard to see that calibration is a major issue with this type of processor.

Processes make use of heuristics to at least start them going. The designer, when invoking any of the decision-making methods, will almost certainly not select the same rules, processes, and parameters that another designer will choose. There is a lack of consistency in AI processors. The outcomes are fuzzy, not crisp as in instrumentation that complies with a physical law. They act like humans do in that they provide a range of solutions to the same problem.

At first sight this seems to imply that we should ignore the AI possibilities for they cannot be calibrated according to long-standing metrological practices. However, their performance is often very worthy and will be workable where algorithmic methods are not. This calibration constraint must be considered in terms of human thinking, not so much in terms of physics and mathematical models.

At present, AI processing methods have been well proved in many fields, these tending to be fields in which performance does not need calibration with regard to the standards regime. Examples are the use of fuzzy logic in clothes washing machines to improve the wash by varying the wash cycle parameters, in neural network methods to aid the recognition of banknotes, and in rule-based controllers in industrial process plant controls. Genetic algorithms have been used to schedule power supply and usage in space shuttles — said to be an impossible task by any other known means. These all are typified as being needs where much experience can be devoted to ensuring they work satisfactorily.

They should normally not be the critical processing element in safety-critical situations. They can certainly be used in routine management situations where they can outperform algorithmic methods, but there they should be backed up with conventional alarms. They are often used in off-line plant control where the human and alarms are still the final arbiter. This, however, seems to be only a cautious step in our slow acceptance of new ideas.

The process of calibration here is more akin to that of conducting evaluation and validation. Does the system give the range of outcomes expected in given circumstances? Are the outcomes better than those without the processor? Could it be done as well or better by algorithmic-based processing? Is the speed it gives worth the problems it may bring? Problems in their testing, and thus calibration, are discussed by Sizemore [6]. The issues that need to be considered in the calibration of conventional instrumentation [5] are relevant to the calibration of AI-based processing but need much more care in their execution.

Such questions require consideration of the very same elements of decision theory upon which they are based to test them. They have been set up to think like humans so it is expected they will have to be calibrated and evaluated like humans — that is not at all easy.

At present, the calibration and validation of AI systems are not standardized well enough. This impedes acceptance, but standardization will improve as world knowledge of this relatively new method of processing develops to greater maturity inside instrumentation.

There will be opposition to the use of AI methods, but the performance gain they bring will ensure they are used. The forward-thinking academic measurement community is showing signs of addressing AI signal processing — but it will take time.

References

1. Baker, D., Kohler, D.C., Fleckenstein, W.O., Roden, C.E., and Sabia, R., Eds., *Physical Design of Electronic Systems,* Vol. 4, Prentice-Hall, Englewood Cliffs, NJ, 1972.
2. Jain, A.K. and Mao, J., Artificial neural networks: a tutorial, *IEEE Comput.,* pp. 31–44, 1996.
3. Kaufmann, A., *The Science of Decision Making,* Weidenfeld and Nicholson, London, 1968.
4. Mauris, G., Benoit, E., and Foulloy, L., Fuzzy sensors for the fusion of information, in *Proc. 13th IMEKO Conference,* Turin, pp. 1009–1014, 1994.
5. Nicholas, J., in *Handbook of Measurement and Control,* Vol. 3, Sydenham, P.H. and Thorn, R., Eds., John Wiley and Sons, Chichester, U.K., 1992.
6. Sizemore, N.L., Test techniques for knowledge-based systems, *ITEA J.,* XI (2), 34–43, 1990.
7. Tang, K.S., Man, K.F., Kwong, S., and He, Q., Genetic algorithms and their applications, *IEEE Signal Processing Mag.,* 13(6), 22–37, 1996.
8. Venmuri, *Artificial Neural Networks: Theoretical Concepts,* IEEE Computer Society Press, Los Angeles, CA, 1988.