Advanced Process Control Overview

What is Advanced Process Control?

Over the past 40 years, much have been written about advanced process control; the underlying theory, implementation studies, statements about the benefits that its applications will bring and projections of future trends.

During the 1960s, advanced process control was taken to mean any algorithm or strategy that deviated from the classical three-term, Proportional-Integral-Derivative (PID), controller. The advent of process computers meant that algorithms that could not be realized using analog technology could now be applied. Feed forward control, multivariable control and optimal process control philosophies became practicable alternatives. Indeed, the modern day proliferation of so called advanced control methodologies can only be attributed to the advances made in the electronics industry, especially in the development of low cost digital computational devices (circa 1970). Nowadays, advanced control is synonymous with the implementation of computer based technologies.

It has been reported that advanced process control can improve product yield; reduce energy consumption; increase capacity; improve product quality and consistency; reduce product giveaway; increase responsiveness; improved process safety and reduce environmental emissions. By implementing advanced control, benefits ranging from 2% to 6% of operating costs have been quoted. These benefits are clearly enormous and are achieved by reducing process variability, hence allowing plants to be operated to their designed capacity.

What exactly is advanced process control?
Depending on an individual's background, advanced process control may mean different things. It could be the implementation of feed forward or cascade control schemes, of time-delay compensators, of self-tuning or adaptive algorithms or of optimization strategies. Here, the views of academics and practicing engineers can differ significantly.

We prefer to regard advanced control as more than just the use of multi-processor computers or state-of-the-art software environments. Neither does it refer to the singular use of sophisticated control algorithms. It describes a practice, which draws upon elements from many disciplines ranging from Control Engineering, Signal Processing, Statistics, Decision Theory, Artificial Intelligence to hardware and software engineering.

The Advanced Process Control topics we will be covering in detail are:

Fuzzy Logic

Fuzzy Logic is successfully used in today's process control systems. Fuzzy logic addresses such applications perfectly as it resembles human decision making with an ability to generate precise solutions from uncertain or approximate information. It fills an important gap in engineering design methods left by mathematical and logic-based approaches.

While other approaches require accurate equations to model real-world behaviors, fuzzy design can accommodate the ambiguities of human languages and logics. It provides both an intuitive method for describing systems in human terms and automates the conversion of those system specifications into effective models.

Natural language is full with vague and imprecise concepts such as "Peter is tall," or "It is very hot today." Such statements are difficult to translate into more precise language without losing some of their semantic value: for example, the statement "Peter's height is 175cm" does not explicitly state that he is tall, and the statement "Peter's height is 1.4 standard deviations about the mean height for men of his age in his country" is fraught with difficulties: would a man 1.3999999 standard deviations above the mean be tall? Which country does Peter belong to, and how is membership in it defined?

While it might be argued that such vagueness is an obstacle to clarity of meaning, only the most staunch traditionalists would hold that there is no loss of richness of meaning when statements such as "Peter is tall" are discarded from a language. Yet this is just what happens when one tries to translate human language into classic logic. Such a loss is not noticed in the development of a financial program,
but when one wants to allow for natural language queries, or "knowledge representation" in expert systems, the meanings lost are often those being searched for.

The notion central to fuzzy systems is that truth values (in fuzzy logic) or membership values (in fuzzy sets) are indicated By a value on the range 0 to 1, with 0 representing absolute Falseness and 1 representing absolute Truth.

A fuzzy logic controller is usually given a set of rules which look like:
- if you see A, do X
- if you see B, do Y
- if you see C, do Z.

The fuzzy controller will look at the situation. In one case, it may see something which has a content of all A, B and C, but not 100% of each. For example it may see a situation which resembles 30% of A, 10% of B and 40% of C. It will therefore tell itself that it needs to come up with an output which is 30% of X, 10% of Y and 40% of Z. The combination of these outputs will be its final output.

<u>Advantages of Fuzzy Logic Control</u>
- Useful when we are not very sure of the process model.

- Many commercial packages are available in DCS Systems

Disadvantages of Fuzzy Logic Control

- Prior knowledge is required. If a case is missed, the controller would not work properly.

Model Predictive Control MPC

MPC is widely adopted in the process industry as an effective means to deal with large multivariable constrained control problems. The main idea of MPC is to choose the control action by repeatedly solving on line an optimal control problem. This aims at minimizing a performance criterion over a future horizon, possibly subject to constraints on the manipulated inputs and outputs, where the future behavior is computed according to a model of the plant.

MPC has been used in industry for more than 30 years, and has become an industry standard (mainly in the petrochemical industry) due to its intrinsic capability for dealing with constraints and with multivariable systems. Most commercially available MPC technologies are based on a linear model of the process. For processes that are highly nonlinear, the performance of an MPC based on a linear model can be poor. This has motivated the development of Nonlinear Model Predictive Control (NMPC), where a more accurate (nonlinear) model of the plant is used for prediction and optimization.

Predictive Constrained Control

PID type controllers do not perform well when applied to systems with significant time-delay. Perhaps the best known technique for controlling systems with large time-delays is the Smith Predictor. It overcomes the debilitating problems of delayed feedback by using predicted future states of the output for control. Currently, some commercial controllers have Smith Predictors as programmable blocks. There are, however, many other model-based control strategies have dead-time compensation properties. If there is no time-delay, these algorithms usually collapse to the PID form. Predictive controllers can also be embedded within an adaptive framework.

Multivariable Control

Most processes require the monitoring of more than one variable. Controller-loop interaction exists such that the action of one controller affects other loops in a multi-loop system. Depending upon the inter-relationship of the process variables, tuning each loop for maximum performance may result in system instability when operating in a closed-loop mode. Loops that have Single Input Single Output (SISO) controllers may therefore not be suitable for these types of applications. These types of controllers are not designed to handle the effects of loop interactions.

A multivariable controller, whether Multiple Input Single Output (MISO) or Multiple Input Multiple Output (MIMO) is used for systems that have these types of interactions. A model-based controller can be modified to accommodate multivariable systems. Loop interactions are considered as feed-forward disturbances and are included in the model description. Following SISO designs, multivariable controllers that can provide time-delay compensation and handle process constraints can also be developed with relative ease. By incorporating suitable numerical procedures to build the model on-line, adaptive multivariable control strategies result.

Model-Based Predictive Control

Model-Based Predictive Control technology utilizes a mathematical model representation of the process. The algorithm evaluates multiple process inputs, predicts the direction of the desired control variable, and manipulates the output to minimize the difference between target and actual variables. Strategies can be implemented in which multiple control variables can be manipulated and the dynamics of the models are changed in real time.

Dynamic Matrix Control

Dynamic Matrix Control (DMC) is also a popular model-based control algorithm. A process model is stored in a matrix of step or impulse response coefficients. This model is used in parallel with the on-line process in order to predict future output values based on the past inputs and current measurements.

Statistical Process Control

Statistical Process Control (SPC) provides the ability to determine if a process is stable over time, or, conversely, if it is likely that the process has been influenced by "special causes" which disrupt the process. Statistical Control Charts are used to provide an operational definition of a "special cause" for a given process, using process data.

SPC has been traditionally achieved by successive plotting and comparing a statistical measure of the variable with some user defined control limits. If the plotted statistic exceeds these limits, the process is considered to be out of statistical control. Corrective action is then applied in the form of identification, elimination or compensation for the assignable causes of variation.

"On-line SPC" is the integration of automatic feedback control and SPC techniques. Statistical models are used not only to define control limits, but also to develop control laws that suggest the degree of manipulation to maintain the process under statistical control. This technique is designed specifically for continuous systems. Manipulations are made only when necessary, as indicated by detecting violation of control limits. As a result, savings in the use of raw materials and utilities can be achieved using on-line SPC

Artificial Neural Networks

What is a Neural Network?

An Artificial Neural Network ANN is an information-processing paradigm inspired by the way the brain processes information. ANNs are collections of mathematical models that emulate some of the observed properties of biological nervous systems and draw on the analogies of adaptive biological learning. An ANN is composed of a large number of highly interconnected processing elements that are analogous to neurons and are tied together with weighted connections that are analogous to synapses.

Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well. Learning typically occurs by example through training or exposure to a truth table set of input/output data where the training algorithm iteratively adjusts the

connection weights (Synapses). These connection weights store the knowledge necessary to solve specific problems.

ANNs are applied to control problems where the input variables are measurements used to drive an output actuator, and the network learns the control function. The advantages of ANNs lie in their resilience against distortions in the input data and their capability of learning. They are often good at solving problems that do not have an algorithmic solution or for which an algorithmic solution is too complex to be found.

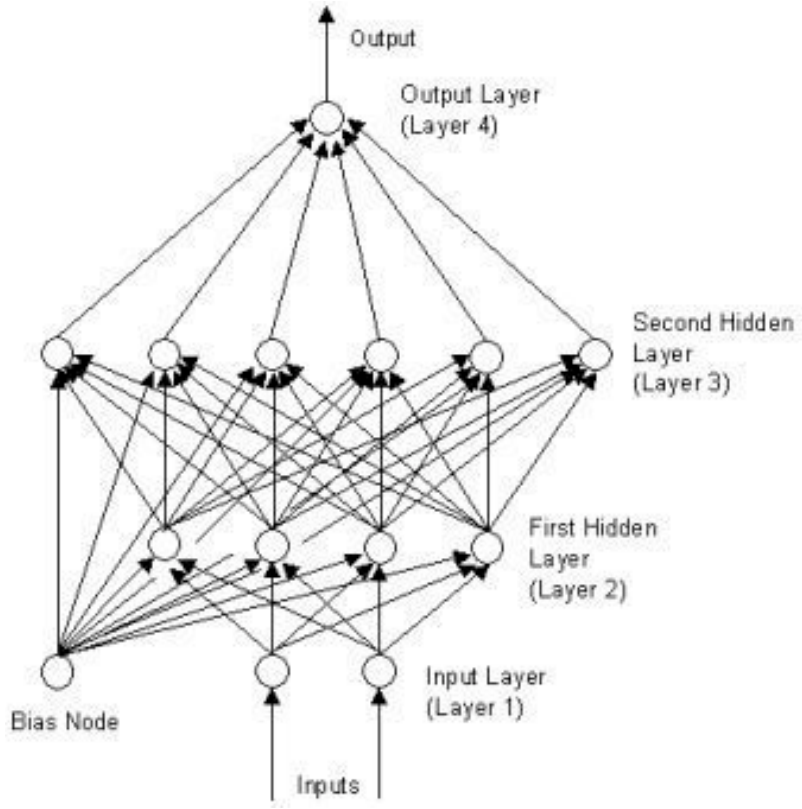

Figure: Typical Feedforward Artificial Neural Network

An Introduction to Neural Networks

Let us first recap the most important features of the neural networks found in the brain. Firstly the brain contains many billions of very special kinds of cell - these are the nerve cells or neurons. These cells are organized into a very complicated intercommunicating network. Typically each neuron is physically connected to tens of thousands of others. Using these connections neurons can pass electrical signals between each other. These connections are not merely *on* or *off* - the connections have varying *strength* which allows the influence of a given neuron on one of its neighbors to be either very strong, very weak (perhaps even no influence) or anything in between. Furthermore, many aspects of brain function, particularly the learning process, are closely associated with the adjustment of these connection strengths. Brain activity is then represented by particular patterns of firing activity amongst this network of neurons. It is this simultaneous cooperative behavior of very many simple processing units which is at the root of the enormous sophistication and computational power of the brain.

Artificial Neural Networks are computers whose architecture is modeled after the brain. They typically consist of many hundreds of simple processing units which are wired together in a complex communication network.

Each unit or node is a simplified model of a real neuron which fires (sends off a new signal) if it receives a sufficiently strong input signal from the other nodes to which it is connected. The strength of these connections may be varied in order for the network to perform different tasks corresponding to different patterns of node firing activity. This structure is very different from traditional computers.

The traditional computers that we deal with every day have changed very little since their beginnings in the 1940's. While there have been very significant advances in the speed and size of the silicon-based transistors, which form their basic elements - the *hardware*, the overall design or *architecture* has not changed significantly. They still consist of a central processing unit or CPU which executes a rigid set of rules (the *program* or *software*) sequentially, reading and writing data from a separate unit - the *memory*. All the "intelligence" of the machine resides in this set of rules - which are supplied by the human programmer. The usefulness of the computer lies in its vast speed at executing those rules - it is a superb machine but not a mind.

Neural networks are very different - they are composed of many rather feeble processing units, which are connected into a network. Their computational power depends on working together on any task - this is sometimes termed *parallel processing*. There is no central CPU following a logical sequence of rules - indeed there is no set of rules or program. Computation is related to a dynamic process of node firings. This structure then is much closer to the physical workings of the brain and leads to a new type of computer that is rather good at a range of complex tasks.

The Hopfield Network

The **Hopfield Neural Network** is a simple artificial network which is able to store certain memories or patterns in a manner rather similar to the brain - the full pattern can be recovered if the network is presented with only partial information. Furthermore there is a degree of stability in the system - if just a few of the connections between nodes (neurons) are severed, the recalled memory is not too badly corrupted - the network can respond with a "best guess". Of course, a similar phenomenon is observed with the brain - during an average lifetime many neurons will die but we do not suffer a catastrophic loss of individual memories - our brains are quite robust in this respect (by the time we die we may have lost 20 percent of our original neurons).

The nodes in the network are vast simplifications of real neurons - they can only exist in one of two possible "states" - firing or not firing. Every node is connected to every other node with some strength. At any instant of time a node will change its state (i.e start or stop firing) depending on the inputs it receives from the other nodes.

If we start the system off with a any general pattern of firing and non-firing nodes then this pattern will in general change with time. To see this think of starting the network with just one firing node. This will send a signal to all the other nodes via its connections so that a short time later some of these other nodes will fire. These new firing nodes will then excite others after a further short time interval and a whole cascade of different firing patterns will occur. One might imagine that the firing pattern of the network would change in a complicated perhaps random way with time. The crucial property of the Hopfield network

which renders it useful for simulating memory recall is the following: we are *guaranteed* that the pattern will settle down after a long enough time to some fixed pattern. Certain nodes will be always "on" and others "off". Furthermore, it is possible to arrange that these *stable firing patterns* of the network correspond to the desired memories we wish to store!

The reason for this is somewhat technical but we can proceed by analogy. Imagine a ball rolling on some bumpy surface. We imagine the position of the ball at any instant to represent the activity of the nodes in the network. Memories will be represented by special patterns of node activity corresponding to wells in the surface. Thus, if the ball is let go, it will execute some complicated motion but we are certain that eventually it will end up in one of the wells of the surface. We can think of the height of the surface as representing the energy of the ball. We know that the ball will seek to minimize its energy by seeking out the lowest spots on the surface -- the wells.

Furthermore, the well it ends up in will usually be the one it started off closest to. In the language of memory recall, if we start the network off with a pattern of firing which approximates one of the "stable firing patterns" (memories) it will "under its own steam" end up in the nearby well in the energy surface thereby recalling the original perfect memory.

The smart thing about the Hopfield network is that there exists a rather simple way of setting up the connections between nodes in such a way that any desired set of patterns can be made "stable firing patterns". Thus any set of memories can be burned into the network at the beginning. Then if we kick the network off with any old set of node activity we are *guaranteed* that a "memory" will be recalled. Not too surprisingly, the memory that is recalled is the one which is "closest" to the starting pattern. In other words, we can give the network a corrupted image or memory and the network will "all by itself" try to reconstruct the perfect image. Of course, if the input image is sufficiently poor, it may recall the incorrect memory - the network can become "confused" - just like the human brain. We know that when we try to remember someone's telephone number we will sometimes produce the wrong one! Notice also that the network is reasonably robust - if we change a few connection strengths just a little the recalled images are "roughly right". We don't lose any of the images completely.

The Perceptron - a network for decision making

An artificial neural network which attempts to emulate this pattern recognition process is called the Perceptron. In this model, the nodes representing artificial neurons are arranged into layers. The signal representing an input pattern is fed into the first layer. The nodes in this layer are connected to another layer (sometimes called the *"hidden layer"*). The firing of nodes on the input layer is conveyed via these connections to this hidden layer. Finally, the activity on the nodes in this layer feeds onto the final output layer, where the pattern of firing of the output nodes defines the response of the network to the given input pattern. Signals are only conveyed forward from one layer to a later layer - the activity of the output nodes does not influence the activities on the hidden layer.

In contrast to the Hopfield network, this network produces its response to any given input pattern almost immediately - the firing pattern of the output is automatically stable. There is no relaxation process to a stable firing pattern, as occurs with the Hopfield model.

To try to simplify things, we can think of a simple model in which the network is made up of two screens - the nodes on the first (input) layer of the network are represented as light bulbs which are arranged in a regular pattern on the first screen. Similarly, the nodes of the third (output) layer can be represented as a regular array of light bulbs on the second screen. There is no screen for the hidden layer - that is why it is termed "hidden"! Instead we can think of a **black box** which connects the first screen to the second. Of course, the magic of how the black box will function depends on the network connections between hidden nodes which are inside. When a node is firing, we show this by lighting its bulb. See the picture for illustration.

We can now think of the network functioning in the following way: a given pattern of lit bulbs is set up on the first screen. This then feeds into the black box (the hidden layer) and results in a new pattern of lit bulbs on the second screen. This might seem a rather pointless exercise in flashing lights except for the following crucial observation. It is possible to "tweak" with the contents of the black box (adjust the strengths of all these internode connections) so that the system can produce any desired pattern on the second screen for a very wide range of input patterns. For example, if the input pattern is a triangle, the output pattern can be trained to be a triangle. If an input pattern containing a triangle and a circle is presented, the output can be still arranged to be a triangle. Similarly,
we may add a variety of other shapes to the network input pattern and teach the net to only respond to triangles. If there is no triangle in the input, the network can be made to respond, for example, with a zero.

In principle, by using a large network with many nodes in the hidden layer, it is possible to arrange that the network still spots triangles in the input pattern, independently of what other junk there is around. Another way of looking at this is that: the network can *classify* all pictures into one of two sets - those containing triangles and those which do not. The perceptron is said to be capable of both *recognizing* and *classifying* patterns.

Furthermore, we are not restricted to spotting triangles, we could simultaneously arrange for the network to spot squares, diamonds or whatever we wanted. We could be more ambitious and ask that the network respond with a circle whenever we present it with a picture which contains *both* triangles, squares but *not* diamonds. There is another important task that the perceptron can perform usefully: the network may be used to draw *associations* between objects. For example, whenever the network is presented with a picture of a dog, its output may be a cat. Hopefully, you are beginning to see the power of this machine at doing rather complex pattern recognition, classification and association tasks. It is no coincidence, of course, that these are the types of task that the brain is exceptionally good at.

A neural network is a powerful data modeling tool that is able to capture and represent complex input/output relationships. The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain. Neural networks resemble the human brain in the following two ways:

A neural network acquires knowledge through learning.

A neural network's knowledge is stored within inter-neuron connection strengths known as synaptic weights.

The true power and advantage of neural networks lies in their ability to represent both linear and non-linear relationships and in their ability to learn these relationships directly from the data being modeled. Traditional linear models are simply inadequate when it comes to modeling data that contains non-linear characteristics.

The most common neural network model is the multilayer perceptron (MLP). This type of neural network is known as a supervised network because it requires a desired output in order to learn. The goal of this type of network is to create a model that correctly maps the input to the output using historical data so that the model can then be used to produce the output when the desired output is unknown.

The MLP and many other neural networks learn using an algorithm called backpropagation. With backpropagation, the input data is repeatedly presented to the neural network. With each presentation the output of the neural network is compared to the desired output and an error is computed. This error is then fed back (backpropagated) to the neural network and used to adjust the weights such that the error decreases with each iteration and the neural model gets closer and closer to producing the desired output. This process is known as "training".

A good way to introduce the topic is to take a look at a typical application of neural networks. Many of today's document scanners for the PC come with software that performs a task known as optical character recognition (OCR). OCR software allows you to scan in a printed document and then convert the scanned image into to an electronic text format such as a Word document, enabling you to manipulate the text. In order to perform this conversion the software must analyze each group of pixels (0's and 1's) that form a letter and produce a value that corresponds to that letter. Some of the OCR software on the market use a neural network as the classification engine.

Of course character recognition is not the only problem that neural networks can solve. Neural networks have been successfully applied to broad spectrum of data-intensive applications, such as:

- Process Modeling and Control - Creating a neural network model for a physical plant then using that model to determine the best control settings for the plant.

- Machine Diagnostics - Detect when a machine has failed so that the system can automatically shut down the machine when this occurs.

- Portfolio Management - Allocate the assets in a portfolio in a way that maximizes return and minimizes risk.

- Target Recognition - Military application which uses video and/or infrared image data to determine if an enemy target is present.

- Medical Diagnosis - Assisting doctors with their diagnosis by analyzing the reported symptoms and/or image data such as MRIs or X-rays.

- Credit Rating - Automatically assigning a company's or individuals credit rating based on their financial condition.

- Targeted Marketing - Finding the set of demographics which have the highest response rate for a particular marketing campaign.

- Voice Recognition - Transcribing spoken words into ASCII text.

- Financial Forecasting - Using the historical data of a security to predict the future movement of that security.

- Quality Control - Attaching a camera or sensor to the end of a production process to automatically inspect for defects.

- Intelligent Searching - An internet search engine that provides the most relevant content and banner ads based on the users' past behavior.

- Fraud Detection - Detect fraudulent credit card transactions and automatically decline the charge.