1. Adaptive Filtering

Adaptive filters represent a significant part of the subject of statistical signal processing upon which they are founded. Historically, the parametric approach has been the main engineering approach to signal processing and is based on *a priori* models derived from scientific knowledge about the problem. At the other extreme, the alternative nonparametric approach is based on the use of more general models trained to replicate desired behaviour using statistical information from representative data sets. Adaptive filters are actually based on an approach which is somewhere in between these two extremes. When *a priori* knowledge of a dynamic process and its statistics is limited then the use of adaptive filters can offer performance improvements over the more conventional parametrically based filter designs. Furthermore, they can offer other signal processing benefits that would not be possible otherwise. Consequently, adaptive filters have found application in diverse fields including communications, controls, robotics, sonar, radar, seismology and biomedical engineering to name but a few.

Filtering in the most general terms is a process of noise removal from a measured process in order to reveal or enhance information about some quantity of interest. Any real data or signal measuring process includes some degree of noise from various possible sources. The desired signal may have added noise due to thermal or other physical effects related to the signal generation system, or it may be introduced noise due the measuring system or a digital data sampling process. Often the noise is a wide-sense stationary random process (has a constant finite mean and variance, and an autocorrelation function dependent only on the difference between the times of occurrence of the samples), which is known and therefore may be modelled by a common statistical model such as the Gaussian statistical model. It may also be random noise with unknown statistics. Otherwise, it may be noise that is correlated in some way with the desired signal itself. The so-called filtering problem can be identified and characterised more specifically by the terms filtering, smoothing, prediction (Haykin 1996) and deconvolution (Hayes 1996).

1. Filtering, strictly means the extraction of information about some quantity of interest at the current time t by using data measured up to and including the time t.

2. Smoothing, involves a delay of the output because it uses information extracted both after and before the current time *t* to extract the information. The benefit expected from introducing the delay is more to do with accuracy than filtering.

3. Prediction, involves forecasting information some time into the future given the current and past data at time *t* and before.

4. Deconvolution, involves the recovery of the filter characteristics given the filter's input and output signals.

Filters can be classified as either linear or nonlinear types. A linear filter is one whose output is some linear function of the input. In the design of linear filters it is necessary to assume stationarity (statistical-time-invariance) and know the relevant signal and noise statistics *a priori*. The linear filter design attempts to minimise the effects of noise on the signal by meeting a suitable statistical criterion. The classical linear Wiener filter, for example, minimises the Mean Square Error (MSE) between the desired signal response and the actual filter response. The Wiener solution is said to be optimum in the mean square sense, and it can be said to be truly optimum for second-order stationary noise statistics (fully described by constant finite mean and variance). For nonstationary signal and/or noise statistics, the linear Kalman filter can be used. Very well developed linear theory exists for both the Wiener and Kalman filters and the relationships between them.

When knowledge of the signal and noise statistics is unavailable *a priori* it is still possible to develop a useful filter by using a recursive algorithm to adjust the filter parameters based on the input data stream. This is what an adaptive filter does. If the signal and noise statistics are stationary then the adaptive filter would be expected to eventually converge to the optimum Wiener solution. If they are nonstationary then the adaptive filter tracks them if they vary at a sufficiently slow rate. The adaptation rate must be faster than the rate of change in statistics to maintain tracking. The parameters of an adaptive filter are updated continuously as the data flows through it; therefore the adaptive filter is strictly a nonlinear system. However, it is common to distinguish linear and nonlinear adaptive filters. A linear adaptive filter is one whose output is some linear combination of the actual input at any moment in time between adaptation operations. A nonlinear adaptive filter does not necessarily have a linear relationship between the input and output at any moment in time. Many different linear adaptive filter algorithms have been published in the literature. Some of the important features of these algorithms can be identified by the following terms (Haykin 1996),

1. Rate of convergence - how many iterations to reach a near optimum Wiener solution.

2. Misadjustment - measure of the amount by which the final value of the MSE, averaged over an ensemble of adaptive filters, deviates from the MSE produced by the Wiener solution.

3. Tracking - ability to follow statistical variations in a nonstationary environment.

4. Robustness - implies that small disturbances from any source (internal or external) produce only small estimation errors.

5. Computational requirements - the computational operations per iteration, data storage and programming requirements.

6. Structure - of information flow in the algorithm, e.g., serial, parallel etc., which determines the possible hardware implementations.

7. Numerical properties - type and nature of quantization errors, numerical stability and numerical accuracy.

The filters described so far may be referred to as classical adaptive filters in so far as they draw upon theory and methods extending from classical Wiener filter theory. A nonclassical approach to adaptive filtering is one that does not rely so much on linear modelling techniques. Artificial neural networks, fuzzy logic, and genetic algorithms have come to prominence in more recent years and are described more as learning systems and belong to the family of CI methods. These employ a range of nonlinear learning techniques that are not dependent on such strict assumptions about either the process model or process statistics. Nevertheless, they can still often be adapted in whole or in part by some form of a gradient descent algorithm (Principe *et al* 2000) that attempts to minimise a mean square error function, not unlike the classical adaptive filters.

1.1 Linear Adaptive Filters

A linear adaptive filter system filters a sequence of input data by controlling its adjustable parameters via an adaptive process. The choice of filter structure is a very important part of the system. There are three main types of structures commonly used (Haykin 1996),

1. Transversal structure (tapped delay line) - similar to the linear FIR filter structure.

2. Lattice predictor - a modular structure with a lattice appearance.

3. Systolic array - a parallel computing network ideally suited for mapping important linear algebra computations such as matrix multiplication, triangulation, and back substitution.

Of these the transversal structure, although not necessarily the most efficient, is very successfully employed for many practical systems. It forms the basis of the

Finite Impulse Response (FIR) discrete-time filter (Loy 1988). The terms associated with this filter structure are defined more thoroughly in later Chapters but for now it is sufficient to say that given an input sequence set of discrete real numbers $\{x[n]\}$, where n is an integer index value, the output sequence $y[n]$ of a Mth order FIR filter is defined by Equation 1.1 and depicted in Figure 1.1. The index value n represents the current discrete-time instant, and n - k represents the previous kth instant, i.e., delayed by k instants.

$$y[n] = \sum_{k=0}^{M} b[k]\, x[n-k] \tag{1.1}$$

where:

$b[k]$ are the fixed filter coefficients that define the filter's characteristics.

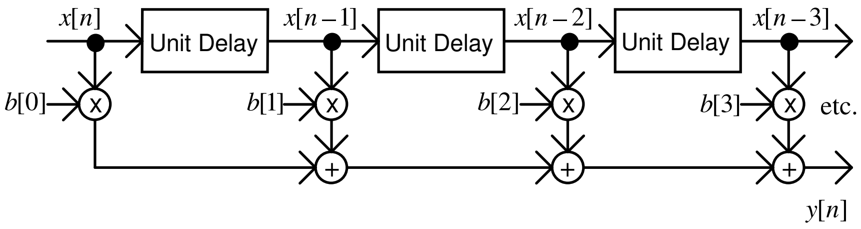

Figure 1.1. FIR Transversal Structure

To design a real FIR filter, Equation 1.1 must be converted into realisable blocks, including a means for obtaining a delayed version of the input sequence $x[n]$, a means for multiplying input signal values in the delay line by the filter coefficients, $b[k]$, and a means for adding the scaled sequence values. The FIR filter is completely defined once the coefficients of the filter are known. For example, if the filter coefficients are the set $\{b[k]\} = \{3, -1, 2, 1\}$ then this represents a third order (M = 3) FIR filter, having a tap length of four. Here, a tap is simply a tap-off point in a serial delay line. The equation for this example filter can be expanded into a four-point (4-tap) difference equation defined by Equation 1.2.

$$\begin{aligned} y[n] &= \sum_{k=0}^{3} b[k]\, x[n-k] \\ y[n] &= b[0]x[n] + b[1]x[n-1] + b[2]x[n-2] + b[3]x[n-3] \end{aligned} \tag{1.2}$$

e.g.,

$$y[n] = 3x[n] + -x[n-1] + 2x[n-2] + x[n-3]$$

The general direct-form realisation of a FIR filter using basic computational elements is depicted in Figure 1.1 with the tapping points shown with black dots. Notice that the input sequence $x[n]$ flows through the delay line continuously and uniformly step by step producing another output value $y[n]$ at each integer step, indefinitely. This filter can be made into an adaptive filter by the addition of a suitable adaptation mechanism that is capable of sensibly adapting the coefficients $b[n]$ progressively at each time step based on some real-time data information.

1.1.1 Linear Adaptive Filter Algorithms

No unique algorithmic solution exists for linear adaptive filtering problems. There are various algorithms and approaches that may be used depending on the requirements of the problem. However, there are two main approaches to the development of recursive adaptive filter algorithms (Haykin 1996),

1. **The Stochastic Gradient Approach** - uses a tapped delay line or transversal structure. The relation between the tap weights and the mean square error between the desired and actual filter output is a multi-dimensional paraboloid (quadratic) error function with a uniquely defined minimum, representing the optimum Wiener solution. This solution can be found by the well-established optimisation method called steepest descent, which uses the gradient vector to gradually descend step by step to the minimum of the error function. The so-called Wiener-Hopf equations, in matrix form, define this optimum Wiener solution. A simpler way to do this is with the Least Mean Squares (LMS) algorithm, invented by B. Widrow and M. E. Hoff Jr in 1959. It is a modified system of Wiener-Hopf equations and is used to adapt the filter weights toward the minimum. This algorithm estimates the gradient of the error function from instantaneous values of the correlation matrix of the tap inputs and the cross-correlation vector between the desired response and the tap weights. The LMS algorithm is very simply and elegantly defined by Equation 1.3.

$$\mathbf{w}[k+1] = \mathbf{w}[k] + 2\eta\, e[k]\, \mathbf{x}[k] \tag{1.3}$$

where:

η	= learning rate parameter.
$e[k]$	= scalar error (desired output minus the actual output).
$\mathbf{x}[k]$	= $[\, x_1, x_2,...., x_p]^T$, the tap vector at time instance k.
$\mathbf{w}[k]$	= $[\, w_1, w_2,..., w_p]^T$, the tap weight matrix at time instance k.

A problem with the LMS algorithm is that it is slow to converge and is dependent on the ratio of the largest to smallest eigenvalue of the correlation matrix of the tap inputs. The higher the ratio, the slower the convergence. Nevertheless, it is very popular and the most widely used learning algorithm, which under the right conditions can perform very adequately. Its tracking behaviour is said to be model-independent and consequently it exhibits good tracking behaviour.

A lattice structure can also be used with the gradient approach in which case the resulting adaptive filtering algorithm is called the Gradient Adaptive Lattice (GAL).

2. **Least Squares Estimation (LSE)** - minimises an objective, or error, function that is defined as the sum of weighted error squares, where the error or residual is defined as the difference between the desired and actual filter output as before. LSE can be formulated in two important ways, with

block estimation or recursive estimation approaches. In block estimation the input data sequence is arranged in blocks of equal time length and processing proceeds block by block. In recursive estimation the processing proceeds sample by individual time sample. Recursive estimation is more popular because it typically requires less data storage overhead than block estimation. Recursive Least Squares (RLS) can be seen as a special case of the well known Kalman filter, which itself is a form of LSE. The Kalman filter uses the idea of state, where state represents a measure of all the relevant inputs applied to the filter up to and including a specific instance in time. In the most general terms the Kalman filtering algorithm can be defined by Equation 1.4.

$$\mathbf{s}[k+1] = \mathbf{s}[k] + \mathbf{K}(k)\,\mathbf{i}[k] \tag{1.4}$$

where:

$\mathbf{K}(k)$	= the Kalman gain matrix at instance k.
$\mathbf{i}[k]$	= the innovation vector at instance k.
$\mathbf{s}[k]$	= the state at instance k.

The innovation vector $\mathbf{i}[k]$ in Equation 1.4 contains the new information (being the observed new data at time k less its linear prediction based on observations up to and including time k-1) that is presented to the filter at the time of the processing for the instance k. As there is a one-to-one correspondence between the Kalman and RLS variables it is possible to learn useful ideas for RLS from the vast Kalman filter literature. There are three main categories of RLS depending on the specific approach taken (Haykin 1996),

1. **The Standard RLS Algorithm** - uses a tapped delay line or transversal structure. Both the RLS and Kalman algorithms rely on the matrix inversion lemma, which results in lack of numerical robustness and excessive numerical complexity. The next two categories address these problems.

2. **Square-root RLS Algorithms** - linear adaptive filter approaches based on QR-decomposition of the incoming data matrix and they represent the square-root forms of the standard RLS algorithm. The QR-decomposition can be performed by the Householder transformation and the Givens rotation, which are both numerically stable and robust data-adaptive transformations.

3. **Fast RLS Algorithms** - by exploiting the redundancy in the Toeplitz structure of the input data matrix (a matrix where all the elements along each of its diagonals have the same value) and through use of linear least squares prediction in both the forward and backward direction the standard and square-root RLS algorithms can be reduced in computational complexity from O(M^2) to O(M), where M is the number of adjustable weights and O(.) denotes "the order of." This reduction in computational

complexity is welcomed for hardware realisations. There are two types of fast RLS algorithms depending on the structure used (Haykin 1996),

1. **Order-recursive Adaptive Filters** - make linear forward and backward predictions using a lattice like structure. These can be realised in numerically stable forms.

2. **Fast Transversal Filters** - where the linear forward and backward predictions are done with separate transversal filters. These suffer from numerical stability problems and they require some form of stabilisation for practical implementations.

The tracking behaviour of the family of RLS algorithms, unlike the LMS algorithm, are model-dependent and therefore their tracking behaviour can be inferior to the stochastic gradient approach unless care is taken to choose an accurate model for the underlying physical process producing the input data.

1.2 Nonlinear Adaptive Filters

The linear adaptive filters discussed above are all based on the minimum mean square error criterion, which results in the Wiener solution for wide sense stationary statistics. This means that these filters can only relate to the second-order statistics of the input data and are strictly only optimum for Gaussian, or at least symmetrical, statistics. It is a fortunate happenstance that these types of filters have been found to be useful for statistics that deviate from this Gaussian ideal. If the input data has non-Gaussian statistics, where the Wiener solution is not guaranteed to be optimum, it is necessary to incorporate some form of nonlinearity in the structure of the adaptive filter to deal adequately with the higher-order statistical information. Although this will improve the learning efficiency it will be at the expense of more complex mathematical analysis of the problem. One important type of nonlinear adaptive filter is the adaptive Volterra filter.

1.2.1 Adaptive Volterra Filters

The adaptive Volterra filter can be seen as a kind of polynomial extension to the linear adaptive filter. It includes a zero order Direct Current (DC) offset term, a first-order linear term, and then a number of higher order terms starting with the second-order quadratic terms, third-order cubic terms and so on to some chosen order. In practice the Volterra filter is often implemented only up to quadratic or cubic order and rarely higher because of the huge increase in computational complexity beyond that, especially for high input dimensions.

1.3 Nonclassical Adaptive Systems

Three types of nonclassical adaptive systems that do not rely on linear modelling techniques are Artificial Neural Networks (ANNs), Fuzzy Logic (FL) and Genetic Algorithms (GAs). All these types of systems can be classified as nonlinear learning structures. Some forms of ANNs are similar to the classical adaptive systems in that they do have a set of parameters that are optimised based on the minimisation of a scalar quadratic error function. To some extent, fuzzy logic systems also have the same kind of similarity as they can be integrated with ANNs to produce hybrid adaptive systems. On the other hand genetic algorithms are different in their form and function although they do have various types of adaptation, or learning mechanisms designed to search, if not for optimal then at least better states.

1.3.1 Artificial Neural Networks

ANNs are a type of massively parallel computing architecture based on brain-like information encoding and processing models. The particular class of supervised training or learning ANNs have a similar external form as the linear adaptive filter. That is, there is a desired output behaviour that the ANN tries to learn as it is exposed to input training data and then it tries to generalise that behaviour after training. In this form ANNs offer the following advantages for adaptive filter applications,

1. Ability to learn the model of almost any continuous (and preferable differentiable) nonlinear physical process given sufficient input-output data pairs generated from that process.

2. Ability to accept weak statistical assumptions about the process.

3. Ability to generalise its learning to new data after initial training.

4. VLSI implementation in a massively parallel structure that is fault tolerant to hardware failure of some of the circuits because of the inherent redundancy.

ANNs, although theoretically able to model linear processes, are less useful for this purpose and should not be used for linear modelling. The well-established linear design methods are easier to use and analyse. The major disadvantage of ANNs is that it is much harder to specify and analyse their application to specific problems. Since the process model is developed from a limited set of training input-output data pairs a degree of uncertainty may exist about the bounds of applicability of the ANN solution. The training data may not be fully representative of the process and may not contain rare but very significant samples that are critical to the system's success.

1.3.2 Fuzzy Logic

Initially fuzzy logic was conceived of as a better method for sorting and handling data but has since proven to be good for control applications because it effectively mimics human control logic. It uses an imprecise but very descriptive language to deal with input data more like a human operator does and it is very robust and forgiving of operator and data input error. FL can be implemented in hardware, software, or a combination of both.

FL is based on the idea that although people do not require precise numerical information input, they are still capable of highly adaptive control functionality. Therefore, it is reasonable to assume that if feedback controllers could be programmed to accept noisy, imprecise inputs, they may be much more effective and perhaps easier to implement. FL uses a simple rule-based approach, such as "IF A AND B THEN C," to control problems as opposed to a strict system model based approach. In that sense a FL model is empirically-based, built by a designer's experience rather than on his/her technical understanding of the system. The design is based on imprecise terms such as "too cool," "add heat," etc. that are descriptive rather than numerically specific. For instance, if a person was trying to regulate the temperature of a shower they might just increase the hot water tap a little if they felt it was too cool and then adjust again if it still was not satisfactory. FL is capable of mimicking this type of behaviour but at a much higher rate than a human can do it.

1.3.3 Genetic Algorithms

Genetic algorithms represent a learning or adaptation method based on search that is analogous to biological evolution and can be described as a kind of simulated evolution. The interest in GAs lies in the fact that evolution is known to be a successful and robust method for biological adaptation. GAs can be seen as general optimisation mechanisms that are not guaranteed to find strictly "optimum" solutions but they often succeed in finding very suitable solutions. They are very useful not only for machine learning problems including function approximation and learning network topologies but also for very many other types of complex problems. In their most common form GAs work with hypotheses that may be described by symbolic expressions, computer programs, specific model parameters, collections of rules, and so on. They are useful in applications where hypotheses contain complex interacting parts, where the impact of each part on overall hypothesis fitness may be difficult to understand or model. GAs can also take advantage of parallel computer hardware since they lend themselves to computational subdivision into parallel subparts.

When the hypotheses are specifically computer programs the evolutionary computing process is called Genetic Programming (GP), where GP is a method for automatically creating computer programs. It starts from a high-level statement of what needs to be done and uses the Darwinian principle of natural selection to breed a population of improving programs over many generations (Koza *et al* 2003). Given a collection or population of initial hypotheses the search for an acceptable hypothesis proceeds from one generation to the next by means of operations inspired by processes in biological evolution such as random mutation and crossover. A measure of "fitness" is required to evaluate the relative worth of the hypotheses in each generation. For each generation the most "fit" hypotheses are selected probabilistically as seeds for producing the next generation by mutating and then recombining their components.

What makes GAs very special is that very little design effort is required to make transitions to new problem solutions within a given domain or even new problems from a completely different domain. In this sense GAs can truly be classified as intelligent in a broader sense because they can provide solutions through a generic approach that can rival solutions produced by human intelligence.

1.4 A Brief History and Overview of Classical Theories

It is both interesting and instructive to consider a brief history of the related areas of linear estimation theory, linear adaptive filters, adaptive signal processing applications and adaptive control. The following historical summary is according to Haykin (Haykin 1996) and the view of Åström and Wittenmark (Åström and Wittenmark 1995). It is not in any sense complete but it is sufficient to provide a suitable structure to relate the fundamentally important discoveries and techniques in these areas. Some, but not all, of the most significant techniques mentioned below are more fully developed and analysed in later Chapters.

1.4.1 Linear Estimation Theory

Galileo Galilei, in 1632, originated a theory of estimation, which he developed to minimise various functions of errors. However, it was Gauss who was given credit for the development of linear estimation theory. This was based on his invention of the method of least squares that he developed in 1795 to study the motion of heavenly bodies. Legendre invented the method of least squares independently of Gauss and actually published before Gauss in 1805 and was therefore subsequently given equal credit for the invention.

In the late 1930s and 1940s Kolmogorov and, Krein and Wiener originated the first studies of minimum mean square estimation in connection with stochastic processes. In 1939 Kolmogorov (Kolmogorov 1939) developed a comprehensive treatment of the linear prediction problem for discrete-time stochastic processes. In 1945 Krein (Krein 1945) subsequently extended Kolmogorov's results to continuous-time by using a bilinear transformation. By 1949 Wiener (Wiener 1949), working independently of either Kolmogorov or Krein, had formulated the continuous-time linear prediction problem but in a different context to the other two. He derived an explicit formula for the optimum predictor as well as solving the filtering problem of estimating a process corrupted by added noise. This required the solution of the integral equation known as the Wiener-Hopf equation, which was developed in 1931.

In 1947 Levinson (Levinson 1947) formulated the Wiener filtering problem in discrete-time in the form of a transversal filter structure as expressed by matrix Equation 1.5.

$$\mathbf{R}\mathbf{w}_0 = \mathbf{p} \tag{1.5}$$

where:

$\mathbf{R}$ = autocorrelation matrix of the tap inputs.
$\mathbf{w}_0$ = tap-weight vector of the optimum Wiener filter solution.
$\mathbf{p}$ = the cross-correlation vector between the tap inputs and the desired output.

For the special case of stationary inputs **R** takes the form of a Toeplitz structure, which allowed Levinson to derive a recursive procedure for solving the matrix Equation 1.5. Later in 1960 Durbin (Durbin 1960) rediscovered Levinson's procedure when he used it for recursive fitting of autoregressive time-series models.

Both Wiener and Kolmogorov assumed that the stochastic process was stationary and that there would be an infinite amount of data available. Other researchers in the 1950s generalised the Wiener and Kolmogorov filter theory for finite observation intervals and for nonstationary processes. However, these solutions were found to be complex and difficult to apply to the prevailing application of satellite orbit estimation. In 1960 Kalman (Kalman 1960) achieved considerable fame with his Kalman filter algorithm, which seemed to be very suitable for the dynamical estimation problems of the new space age. Kalman's original filter was developed for discrete-time processes. A year later in 1961, in conjunction with Bucy, he also developed it for the continuous-time case (Kalman and Bucy 1961).

Over the period from 1968 to 1973 Kailath reformulated the solution to the linear filtering problem by using the so called "innovations" approach, which was first introduced by Kolmogorov in 1941. The term "innovation" conveyed the idea of new information that is statistically independent of past samples of the process, i.e., orthogonal to the linear estimate given all the past data samples.

1.4.2 Linear Adaptive Filters

From earlier work in the 1950s the LMS algorithm for adaptive transversal filters emerged in 1959. It was developed by Widrow and Hoff for their ADALINE pattern recognition system (Widrow and Hoff 1960). The LMS algorithm is a stochastic gradient algorithm and is closely related to the concept of stochastic approximation developed by Robins and Monro (Robins and Monro 1951). The GAL algorithm was developed by Griffiths around 1977 and is only structurally different from the LMS algorithm. In 1981 Zames (Zames 1981) introduced the so called H^∞ norm (or minimax criterion) as a robust index of performance for solving problems in estimation and control. Subsequently, it was shown by Hassibi *et al* (Hassibi *et al* 1996) that the LMS algorithm is optimum under this new H^∞ criterion and thereby proving that its performance is robust. In 1965 Lucky (Lucky 1965) introduced a zero-forcing algorithm alternative to the LMS algorithm for the adaptive equalisation of communication channels, which also used a minimax type of performance criterion.

The family of RLS algorithms saw its beginnings with the work of Placket (Placket 1950). After much work by many researchers, in 1974 Godard (Godard 1974) presented the most successful application of the Kalman filter theory used to derive a variant of the RLS algorithm. It wasn't until 1994 that Sayed and Kailath (Sayed and Kailath 1994) exposed the exact relationship between the RLS algorithm and Kalman filter theory opening the way for the full exploitation of the vast literature on Kalman filtering for solving linear adaptive filtering problems. They showed that QR-decomposition-based RLS and fast RLS algorithms were simply special cases of the Kalman filter.

1.4.3 Adaptive Signal Processing Applications

Five significant applications of linear adaptive signal processing are,

1. Adaptive equalisation.
2. Speech coding.
3. Adaptive spectrum analysis.
4. Adaptive noise cancellation.
5. Adaptive beamforming.

Adaptive equalisation of telephone channels to minimise data transmission intersymbol interference was first developed by Lucky in 1965 (Lucky 1965). He used his minimax criterion based zero-forcing algorithm to automatically adjust the tap weights of a transversal equaliser by minimising what he called the peak distortion. This pioneering work by Lucky spearheaded many other significant contributions to the adaptive equalisation problem. In 1969, Gerosho and Proakis, and Miller independently reformulated the adaptive equaliser problem using a mean square-error criterion. In 1978 Falconer and Ljung (Falconer and Ljung 1978) developed a simplifying modification to a Kalman based algorithm, for adaptive tap adjustment, derived by Godard in 1974. This simplification reduced the computational complexity of Godard's algorithm to that comparable with the LMS algorithm. Satorius, Alexander and Pack in the late 1970s and early 80s showed the usefulness of lattice-based algorithms for adaptive equalisation.

Linear Predictive Coding (LPC) was introduced and developed for the problem of speech coding in the early 1970s by Atal and Hanauer. In LPC the speech waveform is represented directly in terms of time-varying parameters related to the transfer function of the vocal tract and excitation characteristics. The predictor coefficients are determined by minimising the mean square error between actual and predicted speech samples. Although a lattice structure for the linear prediction problem was developed by a number of investigators it was Saito and Itakura who were credited with the invention in 1972. They were able to show that the filtering process of a lattice predictor model and an acoustic tube model of speech were identical.

From the time when Schuster invented the periodogram for analysing the power spectrum of a time-series in 1898 until 1927 it was the only numerical method available for spectrum analysis. In 1927 Yule (Yule 1927) introduced a new approach based on the concept of a finite parameter model for a stationary stochastic process. This new approach was developed to combat the problem of the periodogram's erratic behaviour when applied to empirical time-series observed in nature such as sunspot activity. Yule's model was a stochastic feedback model in which the present sample of the time-series is assumed to consist of a linear combination of past samples plus an error term. This approach was called autoregressive spectrum analysis. Burg rekindled interest in the autoregressive method in the 1960s and 70s with his maximum-entropy method of power spectrum estimation directly from the available time-series. In 1971 Van den Bos (Van den Bos 1971) was able to show that the maximum-entropy method is equivalent to least squares fitting of an autoregressive model to the known autocorrelation sequence. The maximum-entropy method involved the extrapolation of the autocorrelation function of the time series in such a way that the entropy of the corresponding probability is maximised at each step of the extrapolation.

In 1967 Kelly of Bell Telephone Laboratories was given credit for inventing an adaptive filter for speech echo cancellation, which used the speech signal itself in the adaptation processes. Work on echo cancellers only started around 1965. Another type of adaptive noise canceller was the line canceller used for removing the mains power frequency interference from instrument and sensor preamplifier circuits. This was invented by Widrow and his co-workers at Stanford University. An early version of the device was built in 1965 and described in Widrow's paper in 1975 (Widrow *et al* 1975).

Initial contributions to adaptive array antennas were made by Howells in the late 1950s and by Applebaum in 1966. Howells developed a sidelobe canceller that became a special case of Applebaum's adaptive antenna array system. Applebaum's algorithm was based on maximising the Signal-to-Noise Ratio (SNR) at the array output for any type of noise. This classic work was reprinted in the 1976 special issue of IEEE Transactions on Antennas and Propagation (Applebaum and Chapman 1976). Another major work related to adaptive array antennas was put forward independently by Widrow and his co-workers in 1967. Their theory was based on the LMS algorithm and their paper, (Widrow *et al* 1967), the first publication in the open literature on adaptive array antenna systems, was considered to be another classic of that era. In 1969 Capon (Capon 1969) proposed a different method for solving the adaptive beamforming problem based on variance (average power) minimisation. Finally, in 1983, McWhirter (McWhirter 1983) developed a simplified version of the Gentleman-Kung systolic array for recursive least squares estimation, which is very well suited for adaptive beam forming applications.

1.4.4 Adaptive Control

Much of the history that is related to adaptive filters is also relevant to adaptive control systems as they incorporate much of the same theory. In fact, the main difference between the two is mostly a matter of application rather than underlying principles of operation. It is helpful to view signal processing and control theory, generally, as divergent branches of application of the same underlying theory. In some ways there should be more reintegration of the two fields for the sake of economy of understanding.

Historically, adaptive control has been very difficult to define explicitly, because it is seen to be superficially similar to feedback control. Both feedback control and adaptive control involve changing behaviour to conform to new circumstances. Attempts to draw distinctions between the two have not always been successful but it is now commonly agreed that a constant-gain feedback is not an adaptive system. From a pragmatic view point adaptive control can be seen as a special type of nonlinear feedback control in which the states of the process are separated into two categories related to the rate of change involved. In this view the slowly changing states are seen as the parameters and the fast ones are the ordinary feedback states. This definition precludes linear constant parameter regulators and gain scheduling from being called adaptive. Constant parameter regulators do not change their parameters and gain scheduled systems don't have any feedback once the parameters are changed to a new state.

There was extensive research on adaptive control applied to autopilots for high performance aircraft in the early 1950s. The dynamics of high-performance aircraft undergo major changes when they fly from one operating point to another (Levine 1996). This autopilot control problem was investigated by Whitaker *et al* (Whitaker *et al* 1958) using Model Reference Adaptive Control (MRAC). Early enthusiasm for more sophisticated regulators, which work well over a wider range of conditions, diminished through bad hardware, nonexistent theory, a general lack of insight, and finally an in flight test disaster. However, in the 1960s important underlying theory for adaptive control was introduced through the development of state space and stability theory based on Lyapunov and other important results in stochastic control theory. Correct proofs for stability of adaptive systems under very restrictive assumptions were developed in the late 1970s and early 1980s. Nevertheless, controversies over the practicality of adaptive control were still raging, mostly based on the sensitivity and potential instability of earlier designs. From this early work new and interesting research began into the robustness of adaptive control and into controllers that are universally stabilising. By the mid 1980s the field of robust adaptive control was opened based on new designs and analysis. In the late 1980s and early 1990s the focus of adaptive control research turned to extending the results of the 1980s to certain classes of nonlinear plants with unknown parameters. This led to new classes of MRAC with improved transient and steady-state performance.

Adaptive control has traditionally been classified into the MRAC and Adaptive Pole Placement Control schemes (APPC). In MRAC both the poles and zeros of the plant model are changed and in APPC only the poles are changed so that the closed-loop plant has the same input-output properties as those of the reference model.

1.5 A Brief History and Overview of Nonclassical Theories

The three main types of nonclassical adaptive or learning systems are ANN, FL and GAs. These form the foundation of what is now called the computational intelligent systems that have slowly developed into viable and accepted engineering solution methods over the past six decades. Although their origins are not much more recent than the classical adaptive filtering theories they have found broader commercial application only in more recent times.

1.5.1 Artificial Neural Networks

The history of ANNs has two significant periods. The period before 1970 represents the period of initial investigation and the period after 1970 opened the modern era of ANNs.

William James, in 1890, was the first to publish about brain structure and function in connection with psychological theories and neuropsychological research (James 1890). The first theorists to conceive the fundamentals of neural computing were W. S. McCulloch and W. A. Pitts in 1943 (McCulloch and Pitts 1943). They derived theorems related to the then current neural models. Their work proved that networks consisting of neurons could represent any finite logical expression but they did not demonstrate any learning mechanisms. It was Donald Hebb, in 1949, who was the first to define a method of neural network learning (Hebb 1949). Rosenblatt, in 1958, defined the ANN structure called the Perceptron that engineers recognised as a "learning machine" (Rosenblatt 1958). This work laid the foundations for both supervised and unsupervised training algorithms that are seen today in both the Multi-Layer Perceptron (MLP) and Kohonen networks respectively.

The advent of silicon based integrated circuit technology and consequent growth in computer technology in the 1960s was instrumental in the general surge in artificial neural computer systems. The ADALINE introduced by Widrow and Hoff was similar to the Perceptron but it used a much better learning algorithm, called the LMS algorithm, which can also be used for adaptive filters. The extension of the LMS algorithm is used in today's MLP. As the 1960s drew to a close there was growing optimism for the advance of ANN technology. However, funding and research activity in ANNs took a major dive after the publication of Minsky and Papert's book "Perceptrons" in 1969, which was mistakenly thought to have criticised the whole field of ANNs rather than just the simple Perceptron.

The decade of the 1970s saw a much reduced but stable activity in ANN research by a smaller number of researchers including Kohonen, Anderson, Grossberg and Fukushima. After the low period of the 1970s, several very significant publications appeared between 1982 and 1986 that advanced the state of ANN research. John J. Hopfield published a most significant single paper in 1982 (Hopfield 1982) and a follow-on paper in 1984 (Hopfield 1984) identifying ANN structures that could be generalised and that had a high degree of robustness. The Parallel Distributed Processing (PDP) Research Group published the first two volumes of their "Parallel Distributed Processing" in 1986 followed by a third volume in 1988. The most significant contribution of the PDP volumes was the derivation and subsequent popularisation of the Backpropagation-of-error learning algorithm for MLPs. Closely following that, important ANNs based on Radial Basis Functions (RBFs) (Powell 1985) (Broomhead and Lowe 1988) including Donald Specht's Probabilistic Neural Network (PNN) (Specht 1988) and General Regression Neural Network (GRNN) (Specht 1988, 1991) were introduced.

A significant resurgence in interest in ANNs occurred in the 1980s as computers got bigger, faster and cheaper. This ubiquitous computing power allowed the development of many mathematical tools to express analytically, the complex equilibrium state energy landscapes necessary to study ANN architectures. Because of this increased and enthusiastic research activity, especially in conjunction with statistics, many new and useful learning theories have now been proposed and implemented. One of the most important of these is Vapnik's "Statistical Learning Theory" (Cherkassky and Mulier 1998).

1.5.2 Fuzzy Logic

The basic foundations of fuzzy logic were conceived by Lotfi Zadeh in 1965 as an extension of classic set theory (Zadeh 1965). He presented it not as a control methodology, but as a way of processing data by allowing partial set membership rather than specific or crisp set membership/non-membership. Due to inadequacy of computing systems at the time this approach to set theory was not applied to control systems until the 1970s. U.S. manufacturers were not quick to embrace this technology, whereas the Europeans and Japanese began to aggressively build commercial products with it almost from the outset.

Ebraham Mamdani applied FL to control a simple steam engine for the first time in 1974 at the University of London (Mamdani 1974). It was not for another six years that the first industrial application appeared for the control of a cement kiln by F. H. Smidth of Denmark. Fuji Electric of Japan applied FL to the control of a water purification plant in the 1980s and Hitachi later developed an automatic train control system. This led to the FL boom in Japan in the early 1990s with the production of household electronics products using FL. Since then, FL has been applied to a wide range of growing applications including decision support systems, investment consultation, fault diagnosis, medical diagnosis, transport scheduling, management strategy, social and environmental systems (Tanaka 1997).

1.5.3 Genetic Algorithms

In 1948 Alan Turing identified an approach to machine intelligence based on genetical or evolutionary search by which a combination of genes is sought based on survival value. He didn't specify how to conduct the search or mention the concept of population recombination but he did introduce the idea that a number of child-machines should be experimented with to see how well they learn and then choose from the best of these (Turing 1950). Here, the structure of the machine represented hereditary material. Changes of the machine represented mutations, and natural selection (fitness) was based on the experimenter's judgement. It was left to John Holland between 1962 and 1975 to introduce the crucial concepts of maintaining large populations and sexual recombination within them (Holland 1962, 1995).

Since the 1950s there has been a great variety of experimentation with evolution-based computational approaches, which has included optimisation of numerical parameters in engineering design. In 1966 Fogel, Owens and Walsh (Fogel, Owens and Walsh 1966) first developed evolutionary programming, which was a method of evolving finite-state machines. This method was followed up and further developed by numerous researchers including John Koza (Koza 1992). Koza applied the search strategy of GAs to hypotheses consisting of computer programs, which has now come to be known as Genetic Programming (GP).

1.6 Fundamentals of Adaptive Networks

An adaptive network can be used to model either a linear system whose parameters are unknown (or changing with time) or a nonlinear system whose model is unknown (or also changing with time). A linear adaptive system will eventually converge to a linear solution over sufficient time and range of input signals. It will then continue to adapt only if the system or noise statistics change. For a nonlinear process, a linear adaptive system can only adapt to a linear approximation at the current operating point. It is possible however, to keep a historical record of the set of linear models for each small region around a set of operating points and then apply an appropriate model as the set point changes. This is called schedule or switching control with multiple models. A nonlinear adaptive network will adapt to a more accurate model at the current operating point, but like the linear adaptive network it cannot generalise this to new operating points, unless a historical record is kept. To ensure a more robust control of nonlinear systems it is desirable to have some historical information about the system over the expected range of operating points in parallel with a fast adaptive network to make up for any differences.

The basic system structure that is applicable to both adaptive and some learning networks is depicted in Figure 1.2. In the most general terms the vector of the noisy input signal at discrete instance k is $\mathbf{x}_k$ and the vector error signal is $\mathbf{e}_k = (\mathbf{d}_k - \mathbf{y}_k)$, where $\mathbf{d}_k$ is the vector of the noiseless desired output response and $\mathbf{y}_k$ is the actual vector network response given an input of $\mathbf{x}_k$. The network is adapted or trained by presenting it with successive input and desired vector pairs and then using the error signal to adapt the network in such a way as to consistently reduce the error according to some specific learning rule. A least squares error rule is commonly used for both linear networks and nonlinear learning and adaptation. When the network is trained or adapted the error arrives at some statistical minimum. As the statistics of the input vectors change with time the network can be continually adapted to maintain a minimum error, otherwise the network parameters are fixed after training. Either way the network then represents an estimate of the noiseless model of the underlying process or function at that point. The process function is represented by the set of input and desired vector pairs used to train the network.

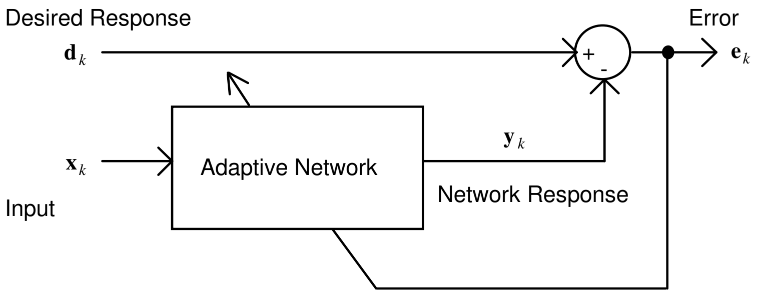

Figure 1.2. Basic Adaptive Structure

There are two main considerations related to the basic adaptive structure shown in Figure 1.2. Firstly, if the desired responses are known why is the network needed at all? Secondly, the adaptation mechanism may be simple or complex depending on the network and consequently, the convergence may take considerable time or computation. In the first case, although the desired responses are not usually known explicitly it is often possible to derive them or find responses that are correlated to them. Since there is no general solution to this problem it is necessary to look at specific examples to gain insight into application issues. The most common generic configurations according to (Lim and Oppenheim 1988) are for,

1. Adaptive prediction.

2. Adaptive forward modelling.

3. Adaptive inverse modelling.

4. Adaptive interference cancelling.

These can best be represented by Figures 1.3 to 1.6 respectively.

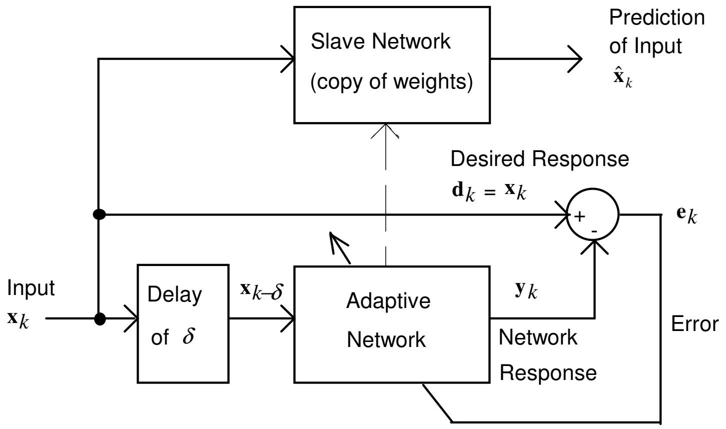

Figure 1.3. Adaptive Prediction

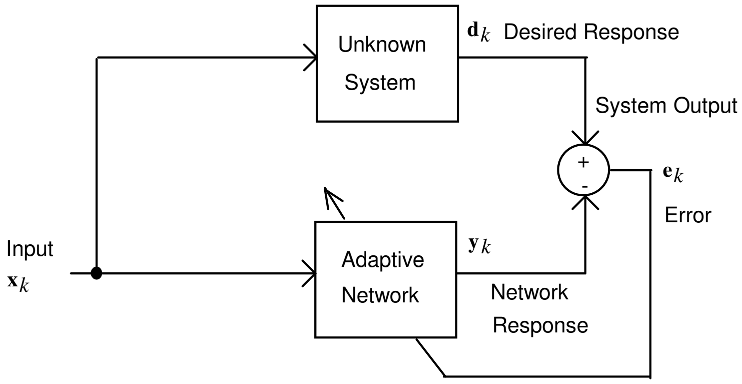

Figure 1.4. System Forward Modelling

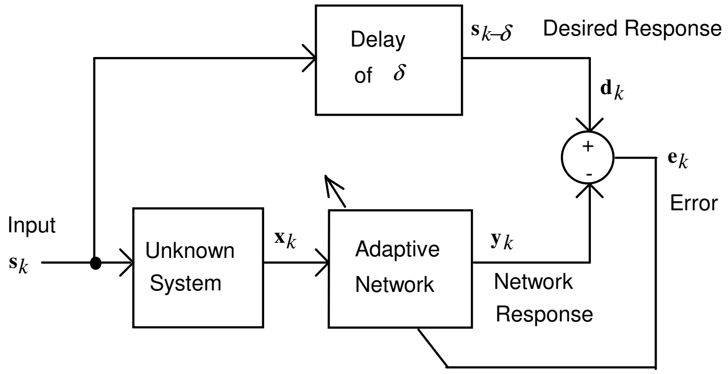

Figure 1.5. Inverse System Modelling

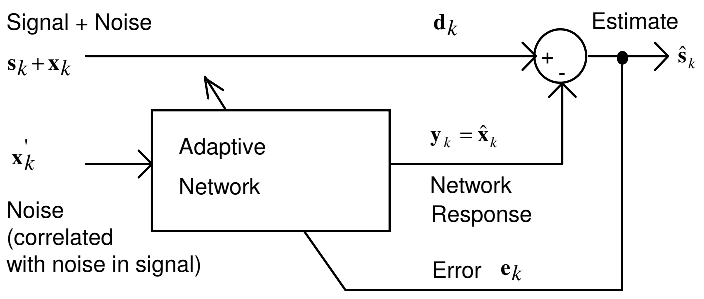

Figure 1.6. Interference Cancelling

For the adaptive prediction model, shown in Figure 1.3, the input is delayed by δ time units and fed to an adaptive network. The input serves as the desired response. The network weights are adapted and when they converge this produces a best estimate of the present input given an input delayed by δ. After convergence the weights are copied into a slave network, which is then taken to be the best predictive model. Wiener developed optimum linear least squares filtering techniques for linear signal prediction. When the signal's autocorrelation function is known, Wiener's theory yields the impulse response of the optimum filter. The autocorrelation function can be determined using a correlator, or alternatively the predictive filter can be determined directly by adaptive filtering. For nonlinear problems and for non-Gaussian noise it is strictly necessary to use adaptation with a nonlinear network to achieve acceptable results.

In cases where a system of unknown structure has observable inputs and outputs an adaptive network as shown in Figure 1.4 can be used to model the system's response. This is called forward system modelling. Inverse modelling involves developing a filter that is the inverse of the unknown system, as shown in Figure 1.5. The delay by δ units is usually included to account for the propagation delay through the plant and the adaptive network, assuming that both are causal systems, i.e., their output depends only on inputs up to and including the current time.

Separating a signal from additive noise, also called interference cancelling, is a common problem in signal processing. An adaptive network as shown in Figure 1.6 can be used to subtract the noise out of the signal. This gives a better result than applying an optimum Kalman or a Wiener filter, both of which introduce some inevitable phase distortion. The adaptive network solution is only viable when there is an additional reference input $\mathbf{x}'_k$ containing noise that is correlated with the original corrupting noise $\mathbf{x}_k$. The network filters the reference noise $\mathbf{x}'_k$, to produce an estimate $\mathbf{y}_k$, of the actual noise $\mathbf{x}_k$. Then, it subtracts $\mathbf{y}_k$ from the primary input ($\mathbf{s}_k$ + $\mathbf{x}_k$), which acts as the desired response $\mathbf{d}_k$. The error signal $\mathbf{e}_k$ becomes the estimate of the signal $\mathbf{s}_k$ if, $\mathbf{s}_k$, $\mathbf{x}'_k$, $\mathbf{x}_k$ and $\mathbf{y}_k$ are statistically stationary, have zero means, and $\mathbf{s}_k$ is uncorrelated with $\mathbf{x}'_k$ and $\mathbf{x}_k$.

1.7 Choice of Adaptive Filter Algorithm

In the most general terms an adaptive algorithm tries to minimise an appropriate objective or error function that involves the input, reference and filter output signals. An objective function must be non-negative and ideally have an optimum value of zero. The adaptive algorithm can be seen to consist of three main parts, the definition of the minimisation algorithm, the definition of the objective function and the definition of the error signal (Dinz 1997).

The most commonly used minimisation methods used for adaptive filters are Newton's method, quasi-Newton methods and the steepest-descent gradient method (Principe 2000). Gradient methods are easy to implement but the Newton method usually requires less iterations to achieve convergence. A good compromise between these two are the Quasi-Newton methods which have reasonable computational efficiency and good convergence. However, the Quasi-Newton methods are susceptible to instability problems. In all these methods the gain or convergence factor must be chosen carefully based on good knowledge of the specific adaptation problem.

The error function can be formed in many ways but the most common ways include the Mean Square Error (MSE), Least Squares (LS), Weighted Least Squares (WLS), and Instantaneous Squared Value (ISV). Strictly speaking the MSE is approximated by the other more practical methods since the MSE is a theoretical value requiring an infinite amount of data. ISV is the easiest to implement but it has noisy convergence properties. The LS method is suitable for stationary data, whereas WLS is valuable for slowly varying data statistics. The choice of error signal is crucial to algorithm complexity, convergence properties, robustness and control of biased or multiple solutions.

There is a great diversity of adaptive applications with their own peculiarities. Every application must be carefully evaluated and understood before a suitable adaptive algorithm can be chosen, because a solution to one application may not be suitable for another. The choice of algorithm must take into account not only the specifics of the application environment but also issues of computational cost, performance, and robustness. Often it can be instructive to apply the simple but robust LMS or Backpropagation-of-error algorithm to the problem first, to study, evaluate and compare the benefits of an adaptive solution to the problem. Further and more detailed design decisions can then be made based on those findings. All adaptive system forms can be implemented to accept and process either real or complex input signals depending on the requirements.