APPLICATIONS OF POLYNOMIAL NEURAL NETWORKS TO FDIE AND RECONFIGURABLE FLIGHT CONTROL[†]

Roger L. Barron, Richard L. Cellucci, Paul R. Jordan, III, and Norman E. Beam, Barron Associates, Inc.

Paul Hess, AbTech Corporation

Andrew R. Barron, Ph.D., University of Illinois

ABSTRACT

Fault detection, isolation, and estimation (FDIE) functions and reconfiguration strategies for flight control systems present major technical challenges, primarily because of uncertainties resulting from limited observability and an almost unlimited variety of malfunction and damage scenarios. This paper deals primarily with a portion of the problem, i.e., global FDIE for single impairments of control effectors. Reference is also made to reconfiguration strategies.

Polynomial neural networks are synthesized using a constrained error criterion to obtain pairwise discrimination between impaired and no-fail conditions and isolation between impairment classes. The pairwise discriminators are then combined in a form of voting logic. Polynomial networks are also synthesized to obtain estimates of the amount of effector impairment. The Algorithm for Synthesis of Polynomial Networks (ASPN) and related methods are used to create the networks, which are high-order, linear or nonlinear, analytic, multivariate functions of the in-flight observables. This paper outlines the design procedure, including database preparation, extraction of waveform features, network synthesis techniques, and the architecture of the FDIE system that has been studied for the Control Reconfigurable Combat Aircraft. Representative performance results are provided.

1. BACKGROUND

1.1 FDIE

Fault detection, isolation, and estimation (FDIE) design for flight control effectors and airframes presents a major technical challenge. Traditional design methods involve specification of all critical design conditions and/or extensive overdesign for unpredicted or uncertain conditions. Only a fraction of the operational fault conditions that might be encountered can be enumerated, and only a small portion of those that may be enumerated can be used explicitly for design, because the number of possible fault conditions is very large. The controlled processes to be dealt with via FDIE models are inherently nonlinear, involve interactions between multiple variables, are high order and time varying, are partially driven by unknown gusts and turbulence, are incompletely observable, and depend upon uncertain aero-inertial parameters and uncertain aeroservoelastic characteristics (particularly for damaged airframes). In advanced systems, over-design will be replaced by real-time adaptation and learning, needed to cope with conditions that cannot be treated explicitly during design.

The control effector, single-failure, global (i.e., based upon airframe responses) FDIE objectives (from Ref. 1):

- FDIE response time of 0.2 sec. or less so as substantially to eliminate acceleration transients and flight-path departures resulting from impairments,
- no false alarms and minimum false dismissals,
- minimum isolation error rate,
- surface impairment estimation within ±10 percent for fractional impairments between 0.2 and 1.0,
- minimum sensitivity to model errors, external disturbances, and measurement noise,
- minimum computational complexity, and
- automated design and evaluation procedures.

Part of the foundation for *single-failure, global FDIE* design is a suitable statistical terminology. The following conditional probabilities are of particular interest:

† The authors acknowledge the sponsorship of the Flight Dynamics Laboratory, Air Force Wright Research and Development Center, Aeronautical Systems Division (AFSC), United States Air Force, Wright-Patterson AFB, Ohio 45433-6553, under Contract F33615-88-C-3615, effective February 22, 1989. The authors express their gratitude to Elisha Rachovitsky (deceased); Thomas J. Molnar, Project Engineer, WRDC/FIGL; and Phillip R. Chandler, WRDC/FIGX, for their technical and administrative guidance in this work.

Invited Paper for presentation at National Aerospace Electronics Conference, Dayton, OH, May 23-25, 1990.

$p\,(\text{det} \mid I_j > T_j)$,
The probability of correct detection given that surface j is impaired by an amount greater than threshold T_j

$p\,(\text{FD} \mid I_j > T_j) = 1 - p\,(\text{det} \mid I_j > T_j)$
The probability of false dismissal given that surface j is impaired by an amount greater than threshold T_j

$p\,(\text{FA})$
The probability of false alarm given that no surface is impaired

$p\,(\text{iso}_j)$
The probability of isolating an impairment to surface j whether or not any impairment exists

$p\,(\text{iso}_j \mid I_j > T_j)$
The probability of correct isolation to surface j given that a correct detection has occurred and that surface j is impaired by an amount greater than threshold T_j.

$p\,(\text{iso}_i \mid I_j > T_j)$ and $p\,(\text{iso}_i \mid I_j \leq T_j)$
The probabilities of false isolation to a surface i other than j given that a correct detection has occurred and that surface j is impaired by an amount greater than (>) or less than or equal (≤) threshold T_j

In a successful FDIE design:

$$p\,(\text{FA}) < 10^{-7} \text{ per flight hour}$$
$$p\,(\text{det} \mid I_j > T_j) \sim 1.0 \text{ for all } j$$
$$p\,(\text{iso}_j \mid I_j > T_j) \sim 1.0 \text{ for all } j$$
$$p\,(\text{iso}_j \mid I_j \leq T_j) \text{ is not important provided } T_j \text{ is small } (\leq 0.2)$$
$$p\,(\text{iso}_j \mid I_i > T_i) \sim 0 \text{ for all } i \neq j$$
$$p\,(\text{iso}_j \mid I_i \leq T_i) \sim 0 \text{ for all } i \neq j$$
$$\sigma_j \leq 0.1 \text{ (10 percent)}$$

where σ_j is the standard deviation of estimation errors for surface j given that a correct isolation to surface j has occurred and that surface j is impaired by an amount greater than threshold T_j.

The false-alarm probability is a function of threshold levels, as are the other probabilities. It is appropriate to describe the behavior of detection and isolation processes in terms of their "operating characteristic" curves, which are graphs of p(det), p(iso), and p(FA) presented as functions of the threshold levels. In communications systems, these are known as "receiver operating characteristic" (ROC) curves. An example of operating characteristic curves for the simulated Control Reconfigurable Combat Aircraft (CRCA) is shown in Figure 1. For greater clarity of p(FA) presentations at large threshold levels, the p(FA) curve is often plotted as log p(FA) vs. a threshold value.

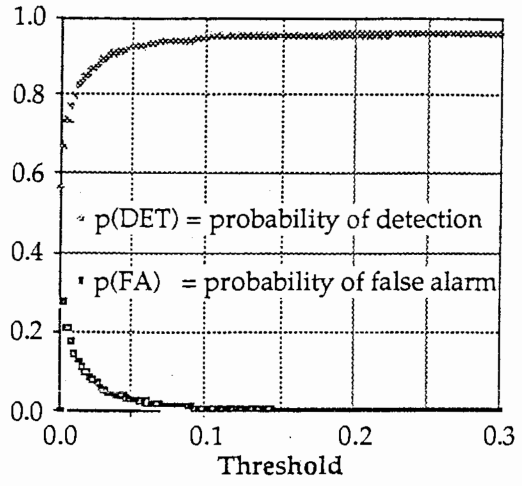

Figure 1: Operating Characteristic Curves for Left Canard Detector/Isolator

Over-all system performance is best summarized by an accuracy ("confusion") matrix of the form shown in Table 1 for the CRCA. The notation ILW means that an effector impairment is isolated to the left wing, IRW the right wing, and so forth.

True Class	Decision Reached (Fraction of Cases)													
	DISMISS	DETECT	ILW	IRW	ICL	ICR	ITE2L	ITE1L	IELL	ITE1R	ITE2R	ITE2R	IRUD	Sum of Off-Diagonal Decisions
No-Fail	*													
Impaired		*												
Left Wing Effector Impaired			*											
Right Wing Effector Impaired				*										
CL Impaired					*									
CR Impaired						*								
TE2L Impaired							*							
TE1L Impaired								*						
ELL Impaired									*					
ELR Impaired										*				
TE1R Impaired											*			
TE2R Impaired												*		
RUD Impaired													*	
Sum of Off Diagonal Decisions														

* Diagonal terms are ideally unity.

Table 1: Performance Matrix (CRCA)

A practical problem arises in determining the operating characteristic curve for p(FA) for relatively large threshold settings: the number of simulation runs required to determine accurately this almost-zero probability can become prohibitively large. The usual compromise solution in communications is to determine p(FA) for several relatively small values of the threshold(s), then extrapolate the p(FA) curve to the region of interest. The system user must be advised that this p(FA) is an estimate until further confirmation becomes available.

Terminology for describing surface impairment estimator performance is presented in Figure 2.

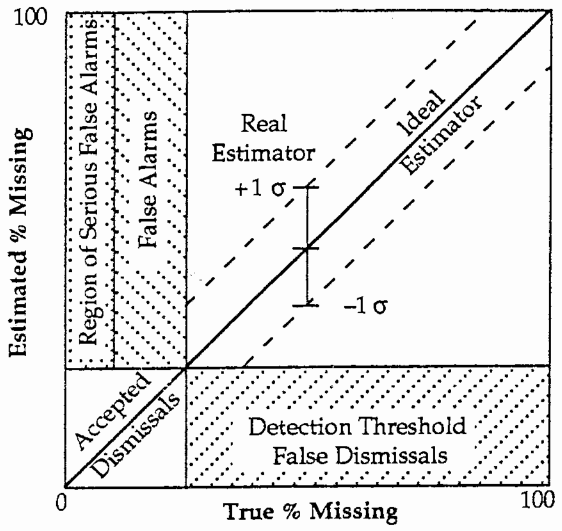

Figure 2: Detector/Estimator Performance Definitions (Based upon Estimator Scatter Plot)

1.2 Reconfiguration

Several fundamental forms of control law reconfiguration are:

(1) *Use of estimated effector sensitivities to adjust control law gains via an explicit pseudo-inverse calculation.*

(2) *Use of estimated effector sensitivities to adjust control law gains via an implicit (neural network) calculation* -- The pseudo-inverse solution may not incorporate fully the designer's wishes for tailoring the gain vector as a more general function of the estimated effector sensitivities. If the designer stipulates the desired gains vs. estimated sensitivities in the form of a numerical database, a polynomial neural network can be synthesized to model these gains.

(3) *Use of fixed, implicit (polynomial network) calculations to infer control law gains independently of estimated effector sensitivities* -- Fixed polynomial neural networks may be used to estimate optimum values of control law gains (independently of estimated effector sensitivities) and/or to predict control system (including aircraft) responses. To design these fixed networks, the designer tabulates, as functions of flight system observables, the optimum gain values and/or correct predictions. In general, these vary with flight, fault, maneuver, and disturbance conditions. Systems using a polynomial network or networks to estimate optimum gains and adjust these in the control law have been called supervisory controllers or *supercontrollers*.

(4) *Use of implicit on-line adaptive (polynomial network) calculations to command the control effectors* -- For severe, multiple effector damage or severe airframe damage, polynomial networks can use unsupervised, recursive on-line adaptation. Thus networks can accomplish implicit identifications of gradually or rapidly varying system dynamics.

2. THE POLYNOMIAL NEURAL NETWORK APPROACH

2.1 Condensed History

Serious study of artificial neural networks began with the work of McCulloch and Pitts (1943), Ref. 2, who put forward a mathematical representation, based upon Boolean algebra, for the gross behavior of neural networks. Hebb (1949), Ref. 3, introduced neural network models in which the cells included delays and a refractory period, and the networks of these cells incorporated feedback connections producing reverberating chains. Lee (1952), Refs. 4, 5, 6, proposed generalized logic-learning elements for automata with selective reinforcement of randomly active parameters in networks of these elements. Farley and Clark (1954), Ref. 7, suggested use of linear elements with thresholds on their outputs. Gabor (1954), Ref. 8, proposed a filter that could learn with supervision. Kolmogorov (1957), proved an important theorem (discussed in Ref. 9) on network representation of continuous functions. Rosenblatt (1957), Ref. 10, drew a vital connection between studies of neural networks and of statistical inference; he showed that a network of the Farley and Clark elements can find a data-separating hyperplane if one exists. (Also see Minsky and Papert, Ref. 11.) Rosenblatt (1958) assembled hardware for a transformation ("Perceptron") network of these elements, using it to recognize patterns. Gabor et al. (1959), Ref. 12, described results obtained with the "universal non-linear filter" which optimized itself by a learning process. The Gabor filter, implemented in hardware, could compute "...94 terms of a polynomial, each term containing products and powers of the input quantities, with adjustable coefficients, and...form their sum". He showed that "...the most general functional of the past of a band-limited time function can be put in the form...of a polynomial of the samples". Widrow (1962), Refs. 13, 14, used a form of stochastic approximation to estimate sequentially the weights in a network of the Farley and Clark elements.

Independently of Gabor, (R. L.) Barron, Gilstrap, et al. (1963; see Refs. 15 - 23) introduced analytic nonlinear neural networks that used polynomial nodal elements suggested by the earlier work of Lee. It was demonstrated that these elements and networks thereof could perform Boolean logic calculations, estimate values of unknown variables, perform high-order predictions, and discriminate between patterns. Pre-structured networks were used; the coefficients simultaneously using a statistical measure of performance on the training data base. The first application was to prediction of re-entry trajectories (1964), Ref. 15. Later in the 1960's, Ivakhnenko, Refs. 24, 25, showed that polynomial networks could be evolved adaptively from a single nodal element to networks of requisite form and complexity. He fitted the network element by element, using a cross-validation procedure (multiple data sets) to select the most promising elements and to detect and avoid overfitting of the data. In 1967, Specht, Ref. 26, examined the generation of polynomial discriminant functions.

Beginning in 1981, (A.R.) Barron systematically developed the connection between neural network synthesis and the tools of statistical inference. First, drawing on the work of Akaike (Ref. 27), Mallows (Ref. 28), and Rissanen (Ref. 29), he derived the predicted squared error (PSE) modeling criterion, Refs. 23, 30-32. PSE is particularly suited to synthesis of estimation networks such as those used in FDIE to estimate effector impairments. In Ref. 32 (A.R.) Barron and (R.L.) Barron compare (1) algorithms for creating neural networks with adaptively synthesized structures; (2) the projection pursuit algorithm of Friedman et al. (1974, 1981, 1984), Refs. 33-35, which is widely used in statistical inference; (3) algorithms for additive models and transformations; and (4) generalizations of these. In Ref. 36, (A.R.) Barron (1989) examines the convergence properties of statistically synthesized neural networks.

In 1982, a paper by Hopfield (Ref. 37) enjoyed wide readership and re-energized worldwide interest in neural networks. Many publications have since appeared (see, for example, Rumelhart et al., Ref. 38, p. 321) that give the impression that multilayer search strategies for synthesis of networks are novel to the 1980s. However, such strategies date to the early 1960s. Recent publications by Giles and Maxwell, Ref. 39, have re-addressed polynomial network principles, referring to "high-order neural networks".

Work began in 1984 to synthesize super-controllers using neural network synthesis algorithms. (See R. L. Barron, Ref. 40; Goldschmidt, R. L. Barron, and Elder, Ref. 41; Elder and R. L. Barron, Refs. 42, 43; and Gross and Migyanko, Ref. 44.) Current work is establishing FDIE neural network design tools and demonstrating their application using the CRCA six-degree-of-freedom nonlinear simulation as a test bed.

The *Algorithm for Synthesis of Polynomial Networks (ASPN)*, Ref. 46, is a culmination of the work of Gabor, (R. L.) Barron, Gilstrap, Ivakhnenko, (A. R.) Barron, Elder, Cellucci, and others toward polynomial neural network synthesis.

2.2 Key Principles

The following important principles have been learned over the past 47 years about the syntheses and applications of neural networks:

(1) Nodal elements should be analytic, and incorporate cross-products as well as sums of their inputs.

(2) Network structures should evolve from simplest forms to levels of just-sufficient complexity as syntheses proceed.

(3) Excessive complexity of neural networks must be avoided to prevent overfitting and consequent unreliability when processing new data.

(4) Global searches are indispensable in assuring network optimality.

(5) Information theoretic (statistical inference) criteria, including overfit penalties, should be used to govern network syntheses. Different performance criteria apply to estimation-network and discrimination-network syntheses.

2.3 Nodal Elements

A large number of candidate nodal elements for neural networks have been used. Some of the most successful are (Ref. 45):

(a) *algebraic combining element*

$$\begin{aligned} y = {} & \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_1 x_2 \\ & + \theta_5 x_1 x_3 + \theta_6 x_2 x_3 + \theta_7 x_1^2 + \theta_8 x_2^2 \\ & + \theta_9 x_3^2 + \theta_{10} x_1^3 + \theta_{11} x_2^3 + \theta_{12} x_3^3 \\ & + \theta_{13} x_1 x_2 x_3 \end{aligned}$$

(b) *linear summing element*

$$y = \theta_0 + \sum_{i=1}^{N} \theta_i x_i$$

(c) *exponential element*

$$y = \theta_0 + \exp(\theta_1 x + \theta_2)$$

(d) *moving-window average weighted-absolute element*

$$y(t) = \sum_{j=0}^{N} \theta_j \, | y(t - j \Delta t) |$$

In the above, y is the element output, the θ's are constant parameters determined during synthesis, and the x's are element inputs. During network synthesis, some coefficients (θ's) may be set to zero. (This is called *carving*, Ref. 46.)

2.4 Algorithms for Fitting

Numerous algorithms for data fitting in neural network synthesis have been used. The most important of these are summarized in Table 2:

	Off-Line		On-Line	
	Static	Dynamic	Static	Dynamic
Least Squares (LS)	▨	□	□	□
Recursive Least Squares (RLS)	▨	□	▨	□
Recursive Prediction Error (RPE)	▨	▨	▨	▨ *
Levenberg-Marquardt (LM)	▩	□	□	□
Recursive Levenberg-Marquardt (RLM)	▩	□	▩	▩ *
Discrete Levenberg-Marquardt (DLM)	▩	▩	□	□
Guided Random Search (GRS)	▩	▩	▩	▩

Legend:	
□	Not Suitable
▨	Suitable Using PSE Criteria for *Estimators*
▩	Also Suitable Using Logistic-Loss Criterion for *Classifiers*
⬚ *	Require Special Pre- and Post-Processing Operations

Table 2: Algorithms for Data Fitting in Neural Network Syntheses

Syntheses of networks involve preparation of databases, nodal element fitting and tuning, elimination of unneeded terms, selection of fitted nodal elements that provide the best data fit vs. element complexity, stopping of network growth at optimum network depth, and network pruning, tuning, implementation, and evaluation. To avoid overfitting, neural network accuracy *must* be assessed in terms of both goodness of fit and network complexity. This assessment is best made during synthesis by using a statistical fitting criterion. After synthesis, a systematic evaluation of network performance should be made throughout its operating region.

2.5 Criteria for Composition and Selection

A generic statistical criterion for composition and selection of functions in neural network syntheses takes the form (Ref. 36)

$$J = \frac{1}{n} \sum_{i=1}^{n} d\left(y_i, f(\underline{x}_i, \hat{\underline{\theta}})\right) + \frac{k}{n}$$

where J is the criterion value (to be minimized) for a given output variable, n is the number of input-output pairs (data vectors) in the synthesis data base, y_i is the ith value of the output variable as represented in the data base, $\underline{x}_i$ is the ith value of the input vector in the data base, $\hat{\underline{\theta}}$ is the estimate of the optimum parameter vector for the network, and k is the dimensionality of $\hat{\underline{\theta}}$. The loss or distortion function, $d(y_i, z_i)$ can take many forms. Of particular interest are:

(a) *for squared-error loss* (estimation networks)

$$d(y_i, z_i) = \frac{1}{2\sigma^2} (y_i - z_i)^2;$$

σ^2 = network error variance (constant)

(b) *for logistic loss* (classification networks)

$$d(y_i, z_i) = - y_i z_i + \log_e (e^{z_i} + e^{-z_i});$$

$$y_i \in \{-1, +1\}, \quad -\infty < z_i < \infty$$

With the squared-error loss function, J becomes proportional to the *predicted squared error* (PSE), an unbiased estimator of future network performance, defined as (Refs. 30-32)

$$\text{PSE} = \text{FSE} + 2\sigma^2 k/n$$

in which FSE is the average fitting squared error of the network and the term $2\sigma^2 k/n$ is the complexity penalty used to constrain the fit. Nearest-neighbor tests can be used prior to network syntheses to determine conservative values of σ^2. The PSE criterion is recommended for syntheses of *estimation* networks and for preliminary syntheses of classification networks.

It can be shown that the values of the $\hat{\underline{\theta}}$ components after PSE minimization for an element are the same as those obtained after the first step of a Newton or Gauss-Newton coefficient adjustment for this element in the logistic-loss case. However, after the search has converged (typically after about five steps), the element adjusted to minimize constrained logistic loss (maximize the constrained log likelihood) provides a much different transformation than one adjusted to minimize PSE.

The logistic-loss function provides the neural network extension of classical logistic regression. For *classification* (discrimination), the constrained logistic-loss criterion should be used to optimize neural networks that are first synthesized using PSE. When a classification network is pre-trained using the PSE criterion, the output of this network must first be re-scaled so as to be interpreted properly in terms of the logistic-loss criterion.

After a network $\hat{f}(\underline{x})$ is trained using the logistic-loss criterion with $y \in \{-1, +1\}$, then $\hat{f}(\underline{x})$ is a sufficient statistic that estimates the log-odds

$$\hat{f}(\underline{x}) = \log_e \frac{\hat{p}(y = +1 \mid \underline{x})}{\hat{p}(y = -1 \mid \underline{x})}$$

The probability that y = 1 given $\underline{x}$ is estimated to be

$$\hat{p}(y = 1 \mid \underline{x}) = \frac{e^{\hat{f}(\underline{x})}}{e^{\hat{f}(\underline{x})} + e^{-\hat{f}(\underline{x})}}$$

PSE syntheses for estimation functions have been very successful in prior work with feedforward networks. Experiments using the logistic-loss criterion indicate that it provides a major improvement for syntheses of classification functions.

2.6 Database Design

Databases used for neural network syntheses must must take into account (1) the conditions under which the system must operate, (2) the means of observing the physical process, and (3) the nature of the physical system itself. Each of these has a strong bearing on database design.

The conditions under which the system must eventually operate should determine the conditions for which data are obtained and the quantity of data. The properties of the data space should determine how the data in the database are distributed. Consider a physical system that is definable in terms of the equivalent of D descriptors. The "space" of these D descriptors has, at its fringes

2^D corners,

$D2^{D-1}$ edges, and

2D spatial boundaries.

The corners (vertices), the points at which all independent variables have limiting values, are of particular concern, because when D is large, most of the volume of the space is crowded near the corners. For the neural network to provide a good model, in most applications it should perform acceptably in the corner regions as well as in the interior of the space. Thus, when D is large, considerably more samples are needed remote from the center of the space, near the edges and particularly near the corners, than are required near the center of the space. To achieve model quality with a nonlinear network at the corners of the data space comparable to the quality at the center, each corner region ideally should be represented by 2^D *times* the density of samples used to represent the interior of the data space. In practice, this may not be practicable and the user should be cautioned that the neural network is less reliable in the corner (and other fringe) regions than in the middle of the data space.

If single-failure FDIE is assumed, only one effector (if any) has non-zero impairment at a time. For the CRCA, the levels of impairment of nine effectors plus the "no-fail" condition define ten descriptors of the simulation space, and because of symmetry within the aircraft, as few as six important descriptors may suffice, ignoring the time of impairment. If the possible commanded maneuvers comprise pitch, roll, and yaw, these contribute another three descriptors. Thus, there may be a total of nine descriptors in the space of data-generation conditions when flight is always in one mode (such as TF/TA). These nine descriptors correspond to 512 corners, and the data density in the vicinity of each of the corners would ideally be 512 times the density in the interior. It is infeasible to provide this, as it would require 512 x 512 data vectors concentrated in corner regions for every vector in the center of the space. Accordingly, one should take several measures to ensure adequate performance of neural networks in spaces involving many descriptors:

(1) Use as many data vectors in the corner regions as are reasonably possible.

(2) Fit the networks to an artificially enlarged data space, so that the fitting quality is

improved within the space of actual variation.

(3) Use nonlinear terms in the networks sparingly.

(4) Carefully test the networks in the corner regions before judging their performance to be acceptable.

(5) When interrogating the networks operationally, verify that each unknown data vector is within the region of accurate representation by the networks.

In sizing the data base, allowance should also be made for generating a significant further data fraction (at least 20 percent) for design evaluation, which must be performed on an independent (and statistically representative) subset of the data.

Concerning data-vector dimensionality, an aircraft has six second-order translational and rotational freedoms. Of the three position components, only altitude is important. The velocity vector azimuthal heading is unimportant. Thus only six dynamical states are required. Nine control-effector states are required for the CRCA. To detect and identify effector impairments, some of the airframe states should be characterized at more than one time, or on a state rate-of-change basis; in this respect, the translational accelerations are particularly useful. Additional "dimensionality" is added by structural bending, thrust level, fuel and payload status, atmosphere conditions, flaps, spoilers, landing gear, uncertainties in aerodynamic properties, and sensor characteristics. All things considered, the CRCA appears to have a minimum intrinsic dimensionality, N, of about 21, and requires, perhaps, as many as 30 observables in the data vector.

2.7 Status of Network Synthesis Algorithms

The key principles presented in Section 2.2 are violated by most neural network synthesis algorithms in present-day use. The Algorithm for Synthesis of Polynomial Networks (ASPN-II) has been developed to remedy this situation. ASPN-II is a *supervised feedforward off-line* neural network synthesis algorithm. A database of input-output pairs is provided by the analyst, and the algorithm creates a network that models the relationships implicit in the data. The synthesized network evolves from the simplest possible form to a nonlinear series-parallel network having just-sufficient complexity for the specific application. ASPN-II can synthesize multiple outputs and accepts up to 200 candidate inputs and outputs. The ASPN-II Facility does the following:

- Assists the analyst with database preparation.
- Determines the best network structure .
- Determines the best algebraic function to use at each node and optimizes the coefficients.
- Writes source code to implement the network (in Fortran or C, and potentially in Ada).
- Computes statistics of network performance on the synthesis data and predicted performance on unseen data from the same statistical population.
- Expresses the network design in block-diagram form and as a multinomial.

3. DESIGN STEPS FOR POLYNOMIAL NETWORK FDIE SYSTEMS

3.1 Off-Line Syntheses

The off-line syntheses of neural network global FDIE functions involve the following steps:

(1) Prepare closed-loop flight control simulation.

(2) Define candidate observables and dependent variables for neural network modeling.

(3) Prepare databases with a variety of flight conditions, maneuvers, impairments, etc.

(4) Synthesize prediction networks from observables. Merge predictor outputs with database.

(5) Using the data for each pair of classes to be discriminated, assess univariate distinguishability value of each observable for each class pair, and select the best 30 observables.

(6) Use Singular Value Decomposition (SVD) to transform observables from Step 5 into their principal-component vectors, which become the candidate inputs for network syntheses.

(7) Using data subsets from Step 6, synthesize (using the logistic-loss criterion) discrimination polynomial neural networks for voting between each pair of effectors.

8) Using data subsets from Step 6, synthesize (using the PSE criterion) an impairment estimator for each effector. Establish thresholds on the outputs of these networks.

(9) Establish production rules for vote tallying based upon outputs of voting networks from Step 7 with veto provisions using the outputs of estimation networks from Step 8. Fractional voting, summation thresholds, and multi-look temporal averaging should be used. The objective

is to establish optimum performance as described by the accuracy matrix (Table 1) and ROC curves.

(10) Evaluate performance of the FDIE subsystem on the training data, independent evaluation data, closed-loop simulations, and closed-loop experiments on a piloted moving-base flight simulator.

The FDIE subsystem uses multiple polynomial networks that run concurrently. The detection process is viewed as a gate-opener for isolation decisions, and isolation as a gate-opener for estimation.

3.2 On-Line Syntheses

On-line syntheses of neural networks for FDIE are similar to the off-line design of these networks, but at least part of the data for design and evaluation come from real-time sensors rather than from simulations. As the on-line data are received, they may be used to re-train the networks on-line. One approach to fast on-line re-training is to use the network structure established during prior (off-line) syntheses and recursively adjust the coefficients on-line. This keeps computational throughput to a minimum on-line. Section 2.4 mentions several recursive algorithms that allow incoming feature vectors to be processed one at a time, as is appropriate for on-line up-dating.

3.3 Simulations

To obtain simulation data, short "flights" of a few seconds each are made. During each flight, it is appropriate to record several data vectors. The times of sampling should be varied randomly from one flight to the next so as to avoid spurious correlations. Points from the first half-second of flight are not used because the history up to one-half second immediately prior to a sample is part of the candidate input for that sample. From each flight, two samples may be taken at random after 0.5 s. and before an impairment occurs, and two samples at random after the impairment occurs. To characterize transients peculiar to some impairments, a fifth sample is taken at random from 0.0 to 0.1 s. after introducing impairment.

The maneuver, command time, and command amplitude are chosen at random in each flight. Each command lasts for 2.0 s. The commands (and the *ranges* of their amplitudes) are single pitch step (- 0.5 g, + 5.0 g), double pitch step (-0.5 g, +1.5 g), single roll-rate step (±240 deg./s), double roll-rate step (± 210 deg./s.), single yaw-rate step (±3.0 deg./s.), double yaw-rate step (±3.0 deg./s.), and simultaneous pitch and roll-rate steps (5.0 g/ ±240 deg./s.). In each flight the effector that is to be impaired, the magnitude of impairment ($0 < I_j \leq 100$ percent), and the time of impairment (within the "failures can occur" window) are selected randomly.

To model aerodynamic uncertainties, each simulated flight uses different aerodynamic characteristics. This is done by randomly choosing aerodynamic force and moment coefficient multipliers at the beginning of each flight. The resultant aerodynamic forces and moments usually are within ± 10 percent (three sigma) of nominal.

3.4 Candidate Inputs

Candidate inputs for neural network syntheses are computed during database preparation. The synthesis algorithm selects the most relevant of these candidate inputs, usually making a different selection for each network. The candidate inputs may include the current observables (M, h, γ, α, β, φ, P, Q, R, a_x, a_y, a_z, surface deflections), predicted "time-now" states based on past states, previous values of observables, and time-averaged observables.

Predicted variables are highly useful because an impairment may be detected by noting that the aircraft is not responding normally. The predictions of expected (no-fail) motion are compared to the measured aircraft motion to calculate time-varying residuals (the unexpected motion). The predictions may be performed by a neural network, as indicated by Figure 3, where "APN" signifies an adaptively-synthesized polynomial network (neural network). Predictions of α, β, a_x, a_y, and a_z have proven particularly useful for FDIE.

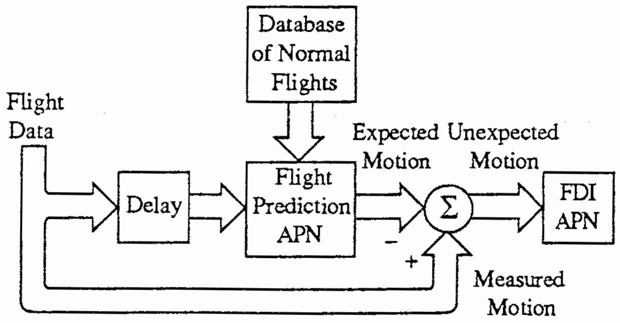

Figure 3: Prediction Network Usage

3.5 Feature Extraction

Once the data vectors are augmented with candidate predicted variables and with previous and back-averaged values of the observables, numerical parameters called *features* are extracted. Feature extraction compresses the information in the data vectors. This reduces network size and can improve performance.

Statistical tools have been prepared that aid feature extraction. Univariate distinguishability testing is used to obtain a preliminary measure of the utility of each of the candidate input variables so that a downselection may be made from a large original list of candidates (typically hundreds) to a shorter list of about 30 promising candidates. In performing this distinguishability testing, the first procedure is to define the *pairwise* discriminations that the FDIE subsystem may be required to make. Usually this definition involves, as a minimum, the set of pairwise discriminations between M + 1 classes of impairment, where M is the number of individual control effectors (nine for the CRCA) and the M + 1st class is the no-impairment (no-fail) class. (Other classes might be defined, such as the group of effectors on the left wing, the group of effectors on the right wing, all effectors, etc., in which case the set of pairwise discriminations would be enlarged, presumably to enhance the decision reliability by means of redundant pairwise discriminations.)

Next, the one-sigma ranges of variation of each of the candidate inputs is computed for each of the classes of impairment (including "no-fail"). One thus obtains a matrix of σ_{ij} values, where i and j represent the candidate input and the impairment class. Now, using a third index, k, to represent any impairment class, the object is to compute a matrix of distinguishability measures for each candidate input (x_i):

$$d_{ijk} = 1 - f_{ijk}$$

wherein

f_{ijk} = fraction of σ_{ij} range of x_i values belonging to class j that overlaps the σ_{ik} range of x_i values belonging to class k

Note that $f_{ijk} = 1$ and $d_{ijk} = 0$ when $j = k$.

These distinguishability measures indicate how much overlap is present along axis i between the data clusters for classes j and k. The overlap is greatest when $d_{ijk} = 0$ and least when $d_{ijk} = 1$. Thus, a value of d_{ijk} close to unity indicates a high degree of separation between the j and k classes if their data clusters are compared along axis i, and a value close to 0 indicates inseparability along axis i.

Generally, the d_{ijk} matrix isn't symmetric, because f_{ikj} doesn't necessarily equal f_{ijk}. So, for M + 1 classes, M (M + 1) distinguishability measures should be examined for each factor x_i, and the candidate inputs for each pairwise discrimination ranked according to their indicated utilities for purposes of discrimination.

Univariate distinguishability testing does not reveal the discrimination utilities of the candidate inputs when these are used nonlinearly and/or combined in multivariate forms. But for each network to be synthesized it is reasonable to work further with the approximately 30 candidates that rank at the top of the univariate utility scale for pairwise discriminations. Likewise, distinguishability tests do not necessarily reveal the most suitable features for nonlinear impairment estimation. It is believed, however, that the best features from the standpoint of distinguishability are also very appropriate for estimation.

Singular value decomposition (SVD) can assist in feature extraction. Applied to the multiple groups of approximately 30 candidate variables (one group for each pairwise discrimination), SVD establishes sets of linear transformations of all the variables in these groups. The top-ranked transformations are usually the most valuable for network syntheses. One may retain as many of the top-ranked transformations for each group as there are intrinsic dimensions in the data. These groups of transformations, along with other variables or features included by the designer, are the candidate inputs for neural network syntheses. The number of candidate inputs should be about the same as the number of intrinsic dimensions of the data space.

3.6 Architecture of the FDIE System

One detection/isolation architecture comprises M(M + 1)/2 pairwise-discrimination (voting) neural networks. These networks should be synthesized using the logistic-loss criterion. Outputs of the voting networks feed into decision logic to determine if a "no-fail" or impaired condition exists and to accomplish impairment isolation. Other networks, synthesized using the PSE criterion, are used to estimate impairments. These estimates are one of the FDIE outputs and may also be incorporated into the decision process as veto provisions.

Using ASPN-II with the PSE criterion (which is intended for estimator syntheses) for syntheses of networks for discrimination between two classes of impairment, it is appropriate to create an estimation task, as shown in Figure 4. This is superior to fitting the network output to –1 for one class and +1 for the other, but still inferior to syntheses with the logistic-loss criterion.

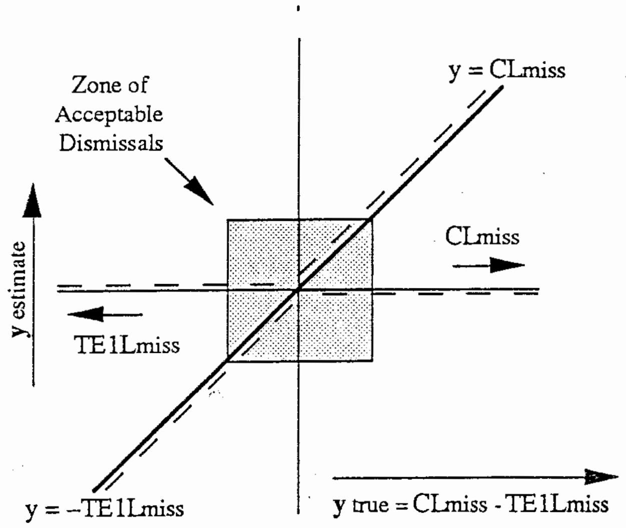

Figure 4: Example Estimation Task for a Class-Discrimination Network

From networks synthesized using PSE, the decision logic may receive tertiary information (below lower threshold, between thresholds, above upper threshold). From networks synthesized using the logistic-loss criterion the information is always the log-odds, and if a threshold is used it must be a single threshold (at zero) used to create a binary vote. The decision logic tallies the votes for all classes. Using integer voting for M + 1 classes (including "no-fail") and M(M+1)/2 pairwise discriminations, M would be the maximum number of votes that can be received for a class, leaving M [(M + 1)/2 – 1] votes scattered among other classes. Often, no class receives M votes. Moreover, it is not unusual to encounter tie votes when using integer voting, and the strategies devised for tie breaking are important determinants of the detection and isolation performance.

Production rules used for vote tallying by the decision logic can have a significant effect on FDIE performance. The primary goals for such rules are to prevent false alarms and minimize incorrect isolations and false dismissals. Fractional voting may be employed, with the more accurate networks given larger vote magnitudes. The estimator networks may be used to veto decisions by the vote-tallying logic in cases of estimated small impairment severity, i.e., within the "acceptable-dismissal" zone.

Some deductive production rules that can used for the CRCA (M = 9) using single-look integer voting include:

(1) If seven, eight, or nine votes are received for "no-fail", then the decision is "dismissal".

(2) If the decision is not "dismissal", and if seven or more votes are received for at least one impairment class, the decision is "detection".

(3) If the decision is "detection", and eight or nine votes are received for only one impairment class, the decision is "isolation (to that class)".

In practice, more sophisticated rules are employed, as discussed in Section 4.

The FDIE system issues a "detection" alarm when it reaches a high level of confidence that an impairment exists. Concurrently, or later, an "isolation" decision is announced once confidence in the same is achieved. When isolation is made to a specific effector, a simultaneous estimate of impairment severity is issued. On the CRCA, it can be difficult to isolate between the three trailing-edge effectors on each wing. If a confident isolation cannot be made to one of these three effectors within a prescribed interval of time, the system announces a "left-wing effector impairment" or a "right-wing effector impairment". If later the specific effector on the wing is isolated, its identity and the estimated severity of its impairment are announced.

The recommended architecture for the global FDIE single-impairment system may thus comprise, for an M-effector aircraft:

(1) M(M + 1)/2 polynomial neural networks used to discriminate between all pairs of hypothesized impairment classes, including "no-fails".

(2) M polynomial neural networks, synthesized using PSE to estimate impairment severity.

(3) Multi-look temporal accumulation of votes.

(4) Vote-tallying logic and production rules.

The designer should seriously consider using one network with M outputs to estimate the multinomial distribution rather than M(M + 1)/2 networks to estimate a Bernoulli (binomial) distribution. The logistic-loss distortion function (Section 2.5) becomes in this case, for data vector $\underline{x}_i$:

$$d(\underline{y}_i, \underline{f}_i) = -\underline{y}_i^T \cdot \underline{f}_i + \log_e\left(1 + \sum_{j=1}^{M} e^{f_{ij}}\right)$$

where the components of $\underline{y}$ are 0, 1 valued, all but one component of $\underline{y}$ being zero ($y_k = 1$ denotes class k); and $f_{ij}(\underline{x}_i, \underline{\theta})$ is the jth component (j = 1, 2..., M) of a vector-valued function $\underline{f}_i$. The objective function (J in section 2) now involves all M + 1 classes (all impairments and "no-fail"), and the synthesis of the M networks involves a Gauss-Newton or other search performed simultaneously on all coefficients of all networks.

Multiple-decision accumulation over time can improve the false-alarm, false-dismissal, and estimation accuracy of the FDIE subsystem. If the sequential decisions and estimates (at, say, a 40 Hz rate) are reasonably uncorrelated, performance is improved (with some reduction in decision speed) by performing temporal confirmation of decisions and temporal averaging of estimates.

4. FDIE ARCHITECTURE FOR CRCA

A convenient first step toward realization of polynomial neural network FDIE systems is to establish the underlying architecture for *single-look* decisions using *integer* voting. This architecture can be refined before proceeding to implementation of a multiple-look (temporal averaging) strategy. Moreover, to facilitate rapid preliminary design, one may wish to begin by using *linear* polynomial classifier networks (and nonlinear impairment estimation networks). The linear classifiers will fall short of the performance of more general classifier networks, but can be synthesized quickly and will establish a baseline performance projection.

The rudder FDIE requires special attention, because this effector remains close to neutral much of the time during simulated maneuvers. It is believed that relatively low rudder activity is typical for many aircraft. When the rudder remains close to neutral, existence of rudder impairment is not readily identifiable from measurements of aircraft response, and estimates of rudder impairment sensitivity are inaccurate.

Ruling out the use of test disturbances, it is wise not to reach a single-look decision or estimate about the rudder unless its measured level of activity is above a reasonable threshold. Moreover, multiple-look decisions and estimates are greatly benefitted from use of only bona fide single-look information. A suitable measure of rudder activity is a moving window of averaged absolute values of the rudder displacement; i.e., one may ask if

$$\sum_i |\delta_r (t - i\Delta t)| \;>\; \text{Threshold.}$$

If this activity level, and/or if commanded yaw rates are below threshold, inferences should not be made about the rudder: it is not appropriate under such conditions to take the votes of the M pairwise networks that involve the rudder (M = 9 for CRCA), and it is also not appropriate to interrogate the rudder impairment estimation network. By excluding the votes and estimates of all of the networks that involve the rudder, the detection and isolation decisions will not be corrupted by M inputs that tend to "tilt" toward their respective non-rudder classes.

It is also found that high roll rates can be confusing to the FDIE system because of aerodynamic and inertial cross-coupling. Some extra data vectors should be devoted to these cases, and, to be conservative, it is best to incorporate special voting rules when the commanded roll rate is large.

Using 36 *linear* pairwise voting networks synthesized using the logistic-loss criterion, with simple vote-tallying rules employed in *single-look* decisions, the detection performance is as shown in Table 3. The nine networks involving the rudder were not allowed to vote, and rudder impairments were not considered in arriving at these results. Using a 40 Hz sampling rate, these single-look decisions are obtained in 25 milliseconds elapsed time of flight.

Minimum Impairment Percentage	Probability of Detection, Percent				
	Canard	TE1	TE2	EL	Total
0	90.9	72.2	64.2	70.3	74.2
25	100.0	92.9	83.7	86.3	90.7
50	99.6	97.6	97.6	96.3	97.7
75	99.1	97.5	100.0	96.8	98.1

Table 3: *Single-Look* Probabilities of Detection Computed on Evaluation Database Using 36 Linear Pairwise Voting Networks

If the sequential single-look detections are statistically independent, the probability of n false alarms in n looks would be

$$p_{n|n}\,(FA) \;=\; [p_{1|1}\,(FA)]^{n}$$

where $p_{1|1}$ (FA) is the probability of a false alarm in a single look. From this equation, again assuming statistically independent decisions,

$$\log [p_{1|1}\,(FA)] = (1/n) \log [p_{n|n}\,(FA)]$$

For $p_{n|n}$ (FA) = 6.9×10^{-13} (equivalent to one false alarm per 10^7 flight hours), and using n = 7, $p_{1|1}$ (FA) = 0.0183 (1.83 percent). Because the sequential decisions are, in fact, highly correlated, 1.83 percent is the allowable *upper* limit on this probability for a false-alarm rate of 10^{-7} per flight hour. For the design existing at the time of this writing, $p_{1|1}$ is approximately 1.0 percent. It is expected that $p_{1|1}$ < 1.0 percent will be achieved with planned refinements.

The 36 linear pairwise voting networks provide canard and wing effector single-look probabilities of detection *and isolation* to the *correct canard* or *correct wing* as shown in Table 4.

Minimum Impairment Percentage	Probability of Detection and Isolation to Correct Canard or Correct Wing, Percent					
	Canard	TE1	TE2	EL	RUD	Total (exc. of RUD)
0	79.7	68.6	60.7	66.1	N/A	68.6
25	94.8	89.3	81.0	82.1	N/A	86.7
50	98.8	92.4	95.3	92.2	N/A	94.5
75	98.2	93.8	97.3	91.0	N/A	94.6

Table 4: *Single-Look* Probabilities of Detection *and Isolation* to Correct Canard or Correct Wing, Computed on Evaluation Database Using 36 Linear Pairwise Voting Networks

The nonlinear single-class estimation networks exhibit the following *single-look* error standard deviations for impairments in the range 0 - 100 percent missing:

Canards,	5.7 percent
TE2 Surfaces,	14.8 percent
TE1 Surfaces,	11.4 percent
Elevators,	17.5 percent
Rudder,	24.5 percent

Rudder estimation accuracy for impairments greater than 45 percent is approximately 20 percent (one sigma).

A scatter plot for the left canard impairment-estimation nonlinear polynomial network (APN) is presented in Figure 8. This plot applies when correct isolation to the respective surface has been made.

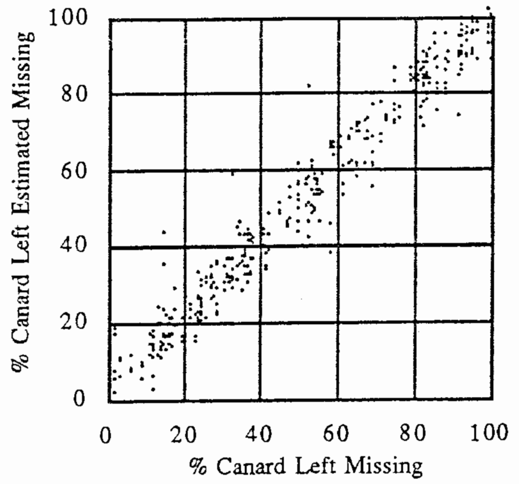

Figure 8: Scatter Plot for Canard Impairment-Estimation Nonlinear Polynomial Network

5. CONCLUDING REMARKS

Polynomial neural networks provide a viable design alternative for FDIE subsystems in reconfigurable flight control architectures. This paper summarizes design principles for these networks. *Single-look* (25 millisecond response time) simulation results are presented that show 86.7 percent probability of detection and correct isolation (to canard or wing) for CRCA effector impairments exceeding 25 percent and 94.4 percent probability of detection and correct isolation for effector impairments exceeding 50 percent. The *single-look* one-sigma impairment estimation accuracy ranges from 5.7 percent for the canards to approximately 20 percent for the rudder. These results are close to the design objectives and will be improved with a *multi-look* strategy. It is expected that the required false-alarm statistics will be achieved using a *seven-look* strategy requiring 0.175 second. Detection probability, false-alarm probability, probability of isolation to the correct effector, and estimation accuracy are expected to equal or exceed design requirements when the seven-look strategy is used. The 0.175 second FDIE response time will also be acceptable and compares very favorably with results from other design approaches.

REFERENCES

1. Barron, R. L., N. E. Beam, R. L. Cellucci, P. R. Jordan, III, P. Hess, and G. J. Montgomery, "Polynomial Network Applications to Global FDIE and Reconfiguration", Barron Associates, Inc. and AbTech Corp. Presentation to WRDC Self-Repairing Flight Control Program Quarterly Review, Wright-Patterson AFB, OH, October 17, 1989
2. McCulloch, W. S., and W. Pitts, "A logical calculus of the ideas immanent in nervous activity", *Bull. Math. Biophys*, Vol. 5, pp. 115-133, 1943.
3. Hebb, D. O., *The Organization of Behavior*, John Wiley & Sons, Inc., NY, 1949.
4. Lee, R. J., "Internal Circuitry of a Reron", privately published, Lib. of Congress Card TK 7882.C5L4, June 13, 1955.
5. Lee, R. J., "Generalization of learning in a machine", *Preprints Papers Presented at 14th Natl. Meeting ACM*, pp. 21-1-21-4, September 1-3, 1959.
6. Lee, R. J., and L. O. Gilstrap, Jr., "Learning machines," *Proc. Bionics Symp.*, USAF Wright Air Development Div., Dayton, Ohio, TR60-600, pp. 437-450, 1960.
7. Farley, B. G., and W. A. Clark, "Simulation of self-organizing systems by digital computers", *IRE Trans. on Inform. Theory*, Vol. PGIT-4, pp. 76-84, 1954.
8. Gabor, D., "Communication theory and cybernetics", *Trans. of IRE*, Vol. CT-1, No. 4, p. 19, 1954.
9. Lorentz, G. G., "The 13th problem of Hilbert," in *Mathematical Developments Arising from Hilbert Problems*, F. E. Browder (Ed.), American Mathematical Society, Providence, RI, 1976.
10. Rosenblatt, F., "The Perceptron", *Cornell Aeronaut. Lab. Rept.* VG-1196-G-1, January 1958.
11. Minsky and Papert, *Perceptrons: An Introduction to Computational Geometry*, M.I.T. Press, Cambridge, MA, 1969.
12. Gabor, D., P.L. Wilby, and R. Woodcock, "A universal non-linear filter, predictor and simulator which optimizes itself by a learning process", *J. IEE*, paper received October 17, 1959.

13. Widrow, B., and M. E. Hoff, "Adaptive switching circuits", *1960 IRE Wescon Convention Record*, pp. 96 - 104, 1960.

14. Widrow, B., "Generalization and information storage in networks of Adeline neurons", *Self-Organizing Systems*, M. C. Yovits, G. T. Jacobi, and G. D. Goldstein (Eds.), Spartan Books, Washington, DC, pp. 435 -461, 1962.

15. Snyder, R.F., R.L. Barron, R.J. Brown and E.A. Torbett, *Advanced Computer Concepts for Intercept Prediction, Vol. I: Conditioning of Parallel Networks for High-Speed Prediction of Re-Entry Trajectories*, Adaptronics, Inc. Technical Summary Rept., Contract DA-36-034-AMC-0099Z, Nike-X Proj. Ofc., U.S. Army Materiel Command, Redstone Arsenal, AL, 1964.

16. Moddes, R.E.J., R.J. Brown, L.O. Gilstrap, Jr., R.L. Barron, et al., *Study of Neurotron Networks in Learning Automata*, Adaptronics, Inc., McLean, VA, AFAL-TR-65-9, February 6, 1965.

17. Gilstrap, L.O., Jr., "An adaptive approach to smoothing, filtering and prediction", *Proc. 1969 NAECON*, pp. 275-280, 1969.

18. Gilstrap, L.O., Jr., "Keys to developing machines with high-level artificial intelligence", ASME Design Engineering Conf., ASME Paper No. 71-DE-21, April 19-22, 1971.

19. Barron, R.L., "Adaptive transformation networks for modeling, prediction, and control", *Proc. Joint Nat'l. Conf. on Major Systems*, IEEE/ORSA, October 25-26, 1971.

20. Mucciardi, A.N., "Neuromime nets as the basis for the predictive component of robot brains", *Cybernetics, Artificial Intelligence, and Ecology* (Robinson, Ed.), Spartan Books, 1972.

21. Barron, R.L., "Theory and application of cybernetic systems: an overview", *Proc. 1974 NAECON*, pp. 107-118, May 1974.

22. Barron, R.L., "Learning networks improve computer-aided prediction and control", *Computer Design*, August 1975.

23. Barron, R.L., A.N. Mucciardi, F.J. Cook, J.N. Craig, and A.R. Barron, "Adaptive learning networks: development and application in the United States of algorithms related to GMDH", *Self-Organizing Methods in Modeling: GMDH Type Algorithms* S.J. Farlow, Ed., Marcel Dekker, Inc., NY, Chap. 2, pp. 25-65, 1984.

24. Ivakhnenko, A.G., "The group method of data handling -- a rival of stochastic approximation", *Soviet Automatic Control*, Vol. 1, pp. 43 - 55, 1968.

25. Ivakhnenko, A.G., "Polynomial theory of complex systems", *IEEE Trans. on Systems, Man, & Cybernetics*, Vol. SMC-1, No. 4, pp. 364-378, October 1971.

26. Specht, D.F., "Generation of polynomial discriminant functions for pattern recognition", *IEEE Trans. on Electronic Computers*, Vol. EC-16, No. 3, pp. 308-319, June 1967.

27. Akaike, H., "Information theory and an extension of the maximum likelihood principle", *Proc. Second Int'l. Symp. on Information Theory*, B.N. Petrov and F. Csaki (Eds.), Akad émiai Kiadó, Budapest, pp. 267 -281, 1972.

28. Mallows, C.L., "Some comments on Cp", *Technometrics*, Vol. 15, pp. 661-675, 1973.

29. Rissanen, J., "A universal prior for integers and estimation by minimum description length", *Ann. Stat.*, Vol. 11, No. 2, pp. 416-431, 1983.

30. Barron, A.R., *Properties of the Predicted Squared Error: A Criterion for Selecting Variables, Ranking Models, and Determining Order*, Adaptronics, Inc., McLean, VA, 1981.

31. A.R. Barron, "Predicted squared error: a criterion for automatic model selection", *Self-Organizing Methods in Modeling: GMDH Type Algorithms* S.J. Farlow, Ed., Marcel Dekker, Inc., NY, Chap. 4, pp. 87-103, 1984.

32. A. R. Barron and R. L. Barron, "Statistical learning networks: a unifying view", *Proc. 20th Symp. on the Interface: Computing Science and Statistics*, Reston, VA, April 1988.

33. Friedman, J.H., "Fitting functions to noisy scattered data in high dimensions", *Proc. 20th Symp. on the Interface: Computing Science and Statistics*, Reston, VA, April 1988.

34. Friedman, J.H., and W. Stuetzle, "Projection pursuit regression", *J. Amer. Stat. Assoc.*, Vol. 76, 817-823, 1981.

35. Friedman, J.H., and J. W. Tukey, "A projection pursuit algorithm for exploratory data analysis", *IEEE Trans. on Computers*, Vol. 23, pp. 881-889, 1974.

36. Barron, A. R., "Statistical properties of artificial neural networks", *Proc. IEEE 1989 Conf. on Decision and Control*, Tampa, FL, December 13 -15, 1989.

37. Hopfield, J., "Neural networks and physical systems with emergent collective computational abilities", *Proc. Nat. Acad. of Sciences*, Vol. 79, pp. 2554 - 2558, 1982.

38. Rumelhart, D. E., G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation", in D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations*, M.I.T. Press, Cambridge, MA, 1986.

39. Giles, C.L., and T. Maxwell, "Learning, invariance, and generalization in high-order neural networks", *Applied Optics*, Vol. 26, No. 23, pp. 4972-4978, December 1, 1988.

40. Barron, R.L., *Identification-based flight control law reconfiguration strategies for effector damage and failures*, Barron Associates, Inc. Task Final Report for Universal Energy Systems, Inc. under Task Order 84-2 of F33615-83-C-3000, Stanardsville, VA, December 6, 1984.

41. Goldschmidt, S.R., R.L. Barron, and J.F. Elder IV, *Super-Controllers: Adaptive Control with APN's*, Barron Associates, Inc. Task Final Report, Part II for Universal Energy Systems, Inc. under Task Order 85-10 of F33614-83-C-3000, Stanardsville, VA, February 28, 1986.

42. Elder, J.F., IV and R.L. Barron, *Uses of Inductive Modeling Tools for Design of Reconfigurable Flight Control Systems*, Barron Associates, Inc. Task Final Report for Century Computing, Inc. under Purchase Order 5238 of F33615-84-C-3609, Stanardsville, VA, July 1987.

43. Elder, J.F., IV and R.L. Barron, "Automated Design of Continuously-Adaptive Control: The 'Super-Controller' Strategy for Reconfigurable Systems", *Proc. 1988 American Control Conference*, June 15-17, 1988.

44. Gross, H.N., and B.S. Migyanko, "Application of super-controller to fighter aircraft reconfiguration", *Proc. 1988 American Control Conference*, June 15-17, 1988.

45. Abbott, D. W., and R. L. Barron, *Active Control of Complex Systems Via Neural Networks Having Internal Feedback Paths and Time Delays*, Barron Associates, Inc. Technical Progress Rept. 1 for Office of Naval Research, Contract N00014-89-C-0137, Stanardsville, VA, August 15, 1989.

46. Elder, J.F., IV, R.L. Cellucci, and R.L. Barron, *Users' Manual: ASPN -- Algorithm for Synthesis of Polynomial Networks, Version 7.70*, Barron Associates, Inc., Stanardsville, VA, 1989.