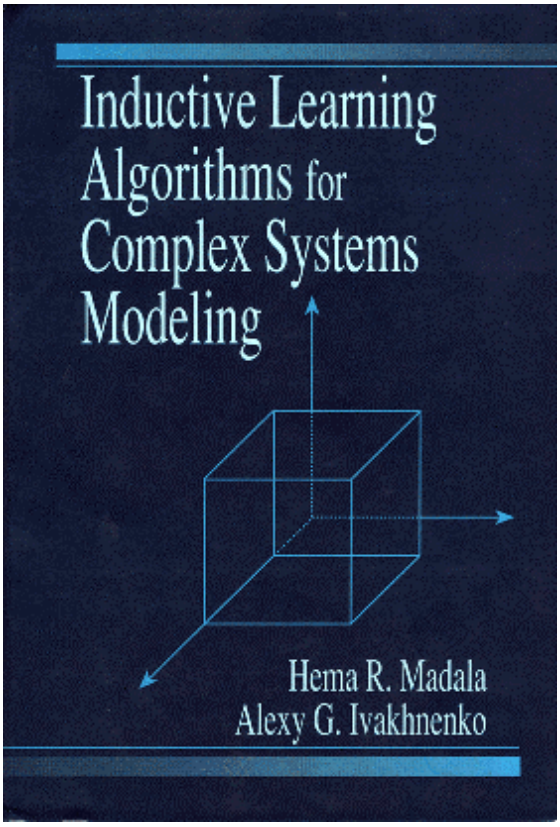

Part from the Chapter 2 of the book:

Madala H.R. and Ivakhnenko A.G.,

"Inductive Learning Algorithms for Complex System Modeling",

1994, CRC Press, ISBN: 0-8493-4438-7.

From abstract:

Inductive Learning Algorithms for Complex Systems Modeling is a professional monograph that surveys new types of learning algorithms for modeling complex scientific systems in science and engineering. The book features discussions of algorithm development, structure, and behavior; comprehensive coverage of all types of algorithms useful for this subject; and applications of various modeling activities (e.g., environmental systems, noise immunity, economic systems, clusterization, and neural networks). It presents recent studies on clusterization and recognition problems, and it includes listings of algorithms in FORTRAN that can be run directly on IBM-compatible PCs.

Inductive Learning Algorithms for Complex Systems Modeling will be a valuable reference for graduate students, research workers, and scientists in applied mathematics, statistics, computer science, and systems science disciplines. The book will also benefit engineers and scientists from applied fields such as environmental studies, oceanographic modeling, weather forecasting, air and water pollution studies, economics, hydrology, agriculture, fisheries, and time series evaluations.

Features:

- Discusses algorithm development, structure, and behavior
- Presents comprehensive coverage of algorithms useful for complex systems modeling
- Includes recent studies on clusterization and recognition problems
- Provides listings of algorithms in FORTRAN that can be run directly on IBM-compatible PCs

Chapter 2
Inductive Learning Algorithms

1 SELF-ORGANIZATION METHOD

The existence of man on earth began about two million years ago. The twentieth century is the turning point for changes, the likes of which have never before occurred in human history. The change in the way of living, modern production processes, scientific-technical revolution, and important changes in and rapid development of computers for information processing and control of complex objects, including the hereditary characteristics of living organisms, constitute but a short list of new phenomena that characterize our age. The expanding possibilities are accompanied by growing anxieties such as how to cope with increasing pollution problems and how to find new food and energy sources in time to feed and warm an increasing population. These and other important problems will be solved in the near twenty-first century—the century in which our children and grandchildren will develop. It is clear why our eyes are turned to the future, to the twenty-first century. We cannot say that we are completely ignorant, because this is not true. For example, we know the fundamental trends of development of scientific-technical progress. We know that this leads to the qualitative mathematical predictions of the future that are possible only with the methods of quantitative mathematical modeling which answer questions about the time periods of the predictions (when?), the forms (how, in which way, and under what conditions?), and the place (where?). Futurology is a young mathematical science whose purpose is to develop methods for the prediction of the future. This branch of science is slowly coming to a mature stage when some of its unavoidable meanderings and errors can be executed. However, we will see here some of its successes and achievements.

A long time ago scientists knew how to calculate exactly the motions of the planets, how to predict eclipses of the moon and of the sun because these are described by exact deterministic equations. Even these were achieved in successful prediction of variable quantities, averaged over a long period of time. Thus, it is not difficult to predict the average temperature of the earth for the entire twenty-first century or to calculate the amount of precipitation which will fall on the mainland of Asia during the same long period of time. The question is, however, how much can the averaging time intervals and area sectors be decreased? How does one obtain predictions for each month and season or year when the prediction interval extends over scores of years? Can scientists actually predict the distant future in general? Such a question usually comes into our minds because of lack of success in many detailed (nonaveraged) quantitative forecasts. The first fiasco was suffered by the so-called probabilistic methods of long-term predictions. Probability is also a type of averaging, and is useful in predictions of the future. Probabilistic methods have their own drawbacks because of their lack of inductive analysis.

Simulation modeling is another area used by Forrester [20] in studying world dynamics. The fact is that simulation modeling does not require any test data. Equations that describe separate parts of the object of prediction are invented by modelers on the basis of their subjective ideas of the object. These models apparently do not take into account the variability of the object characteristics. The common feature of probabilistic and simulation methods is that "the more complex the model, the more accurate" the interval of empirical observations used for estimating the parameters. The modeler cannot tell whether the model is accurate or not in the interval of the object that he lacks the knowledge about. The self-organization modeling that is described here plays an important role in such conditions.

The self-organization method uses a very general meta-language, rather than a language of detailed instructions. The quantitative model built up from the observations should be the same as the model built up from other observations taken at different times and places. This is the prerequisite in obtaining a predictive model through the inductive approach. Let us first go through the basic iterative algorithm based on the inductive approach and the basic network structures that have been in use since the beginning of the usage of these algorithms. Later we will study basic directions, principal characteristics of the algorithms and advanced multilevel achievements in complex problem solving.

1.1 Basic iterative algorithm

In problem solving, the main strategy is to specify a set of proper input-output associations and the main goal is to design an efficient learning algorithm, which is regarded as a search procedure that correctly solves the problem. Learning in the networks takes a variety of forms; mainly it discovers statistical features for detecting regularity. Self-organization is considered while building the connections among the units through a learning mechanism to represent discrete items. The self-organizing process that is established using various heuristics in the network structure helps in obtaining the optimum output response. In the network each unit is represented independently as a black box to generate the input-output relationship as a state function and group of units are treated as a layer of certain thresholding hierarchical stage. The relationships are established through the connecting weights that are estimated by adapting a parameter estimation technique. The measure of the objective function as an "external complement" is used as a threshold value on a competitive basis to make the unit "on" or "off" and if the unit is "on" its output is fed forward to the next layer as inputs. The measure is also considered as an objectivity measure of the unit. Overall, this works as a search in the domain of solution space through a sort of competitive learning. Relevance of local minimum depends on the complexity of the task on which the system is trained by building up heuristics like design of objective function, design of input-output relationships or summation functions at the units, and usage of empirical data for training and testing the network. When one of the units in a particular layer achieves the global minimum of the measure, the processing is stopped and information about the optimal response is recollected from the associated units in the preceding layers. The global minimum is guaranteed because of steepest descent in the output error with respect to the connection weights in the solution space in which it is searched as per a specific objective through cross-validating the weights.

Suppose we have a sample of N observations, a set of input-output pairs (I_1, o_1), (I_2, o_2), $\cdots$, $(I_N, o_N) \in N$, where N is a domain of observations corresponding to the empirical data, and we have to train the network using these input-output pairs to solve two types of problems: (1) identification problem—the given input $I_j (1 \leq j \leq N)$ of variables x corrupted by some noise expected to reproduce the output o_j and to identify the physical laws, if any, embedded in the system; (2) prediction problem—the given input I_{N+1} ex-

pected to predict exactly the output o_{N+1} from a model of the domain studied during the training.

In solving these, a general relationship between output and input variables is built in the form of a mathematical description, which is also called a reference function. Usually the description is considered as a discrete form of the Volterra functional series or Kolmogorov-Gabor polynomial:

$$y = a_0 + \sum_{i=1}^{m} a_i x_i + \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij} x_i x_j + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m} a_{ijk} x_i x_j x_k + \cdots, \tag{2.1}$$

where the output is designated as y, the input vector as $\mathbf{x} = (x_1, x_2, \cdots)$ and $\mathbf{a}$ is the vector of coefficients or weights. This is linear in parameters a and nonlinear in x. Components of the input vector $\mathbf{x}$ could be independent variables or functional terms or finite difference terms. This means that the function could be either an algebraic equation or a finite difference equation, or an equation with mixed terms. The partial form of this function as a state or summation function is developed at each simulated unit and is activated in parallel to build up the complexity.

Unit level

Each simulated unit k receives input variables—for example, $(x_i, x_j) \subset \mathbf{x}$; $i \neq j$—and generates a function $f(\,)$ which is a partial form of the reference function.

$$f(x_i, x_j) = \nu_0^{(k)} + \nu_1^{(k)} x_i + \nu_2^{(k)} x_j + \nu_3^{(k)} x_i x_j + \nu_4^{(k)} x_i^2 + \nu_5^{(k)} x_j^2, \tag{2.2}$$

where $\nu^{(k)}$ are the connecting weights. If we denote o as the desired values and y as the estimated values of the outputs for the function being considered, the output errors would be given by

$$e_p = y_p - o_p; \quad p \in N_A. \tag{2.3}$$

The total squared error for that input vector is

$$E = \sum_{p \in N_A} e_p^2. \tag{2.4}$$

This corresponds to the minimization of the average error E in estimating the weights $\nu^{(k)}$; this is the least squares technique. The weights are computed using a specific training sample N_A which is a part of the whole data points N specified for this purpose.

Layer level

Each layer contains a group of units that are interconnected to the units in the next layer. The weights at each unit are estimated by minimizing the error E. The measure of an objective function is used as the threshold value to make the unit "on" or "off" in comparison with the testing data N_B which is another part of N and, at the same time, it is considered to obtain the optimum output response; i.e., this is used as threshold as well as objectivity measures simultaneously. The outputs of the units which are "on" as per the threshold values are connected as inputs to the units in the next layer; that means that the output of kth unit, if it is in the domain of local threshold measure, would become input to some other units in the next level. The process continues layer after layer. The estimated weights of the connected units are memorized in the local memory.

Functional flow of the algorithm

The flow of the algorithm can be described as follows. We feed the input data of m input variables of x randomly; for example, if they are fed in pairs at each unit, then a total of $C_m^2 (= m(m-1)/2)$ partial functions of the form below are generated at the first layer:

$$y = f(x_i, x_j); \quad i,j = 1,2,\cdots,m; \quad i \neq j, \tag{2.5}$$

where $f(\,)$ is the partial function as in Equation 2.2 and y is its estimated output. Then outputs of $F_1 (\leq C_m^2)$ functions ("freedom-of-choice") are selected as per the threshold measure to pass on to the second layer as inputs in pairs. In the second layer we check the functions of the form

$$z = f(y_i, y_j); \quad i,j = 1,2,\cdots,F_1; \quad i \neq j, \tag{2.6}$$

where $f(\,)$ is the same form of the partial function as in Equation 2.2 and z is its estimated output. The number of such functions is $C_{F_1}^2$. Outputs of F_2 functions are selected to pass on to the third layer. In the third layer we estimate the functions of the form:

$$v = f(z_i, z_j); \quad i,j = 1,2,\cdots,F_2; \; i \neq j, \tag{2.7}$$

where v is the estimated output of the type of function as in Equation 2.2. The number of such functions is $C_{F_2}^2$. The process continues and is stopped when the threshold value begins to increase. The parameters of the optimal function are retrieved through the path of the connecting units from the preceding layers.

2 NETWORK STRUCTURES

The network structures differ as per the interconnections among the units and their hierarchical levels. There are three main inductive learning networks: multilayer, combinatorial, and harmonic. There are other networks based on these three using the concept of self-organization modeling. Multilayer algorithm uses a multilayered network structure with linearized input arguments and generates simple partial functionals. Combinatorial algorithm uses a single-layered structure with all combinations of input arguments including the full description. This could be realized in different ways at each layer of multilayer structure by restricting the number of selected nodes at each layer. Harmonic algorithm follows the multilayered structure in obtaining the optimal harmonic trend with nonmultiple frequencies for oscillatory processes.

2.1 Multilayer algorithm

Multilayer network is a parallel bounded structure that is built up based on the connectionistic approach given in the basic iterative algorithm with linearized input variables and information in the network flows forward only. Each layer has a number of simulated units depending upon the number of input variables. Two input variables are passed on through each unit. For example, x_i and x_j are passed on through kth unit and build a summation function. Weights are estimated using the training set A. At the threshold level, error criterion is used to evaluate this function using the test set B. If there are m input variables, the first layer generates $M_1 (= C_m^2)$ functions. $F_1 (\leq M_1)$ units as per the threshold values are made "on" to the next layer. Outputs of these functions become inputs to the second layer and the same procedure takes place in the second layer. It is further repeated in successive layers until a global minimum on the error criterion is achieved. If it is not

achieved, it means the heuristic specifications must be considered for alteration. The partial function that achieves the global minimum is treated as an optimal model under the given specifications.

The mathematical description of a system can be considered as a nonlinear function in its arguments which may include higher ordered terms and delayed values of the input variables.

$$y = f(x_1, x_2, \cdots, x_1^2, x_2^2, \cdots, x_1x_2, x_1x_3, \cdots, x_{1(-1)}, \cdots, x_{1(-2)}, \cdots), \tag{2.8}$$

where $f(\,)$ is a function of higher degree and y is its estimated output. This can be renotated as a linearized function by calculating all arguments of x in the following form of full description.

$$\begin{aligned} y &= f(u_1, u_2, \cdots, u_m) \\ &= a_0 + a_1u_1 + a_2u_2 + \cdots + a_mu_m, \end{aligned} \tag{2.9}$$

where u_i, $i = 1, 2, \cdots, m$ are the renotated terms of x; a_k, $k = 0, 1, \cdots, m$ are the coefficients and m is total number of arguments. These m input variables become inputs to the first layer. The partial functions generated at this layer are

$$\begin{aligned} y_1 &= \nu_{01}^{(1)} + \nu_{11}^{(1)}u_1 + \nu_{21}^{(1)}u_2, \\ y_2 &= \nu_{01}^{(2)} + \nu_{11}^{(2)}u_1 + \nu_{21}^{(2)}u_3, \\ &\cdots \\ y_{M_1} &= \nu_{01}^{(M_1)} + \nu_{11}^{(M_1)}u_{m-1} + \nu_{21}^{(M_1)}u_m, \end{aligned} \tag{2.10}$$

where M_1 $(= C_m^2)$ is the number of partial functions generated at the first layer, y_j and $\nu_{i1}^{(j)}$, $j = 1, 2, \cdots, M_1$; $i = 0, 1, 2$ are the estimated outputs and corresponding weights of the functions. Let us assume that F_1 functions are selected for the second layer and that there are M_2 $(= C_{F_1}^2)$ partial functions generated at the second layer.

$$\begin{aligned} z_1 &= \nu_{02}^{(1)} + \nu_{12}^{(1)}y_1 + \nu_{22}^{(1)}y_2, \\ z_2 &= \nu_{02}^{(2)} + \nu_{12}^{(2)}y_1 + \nu_{22}^{(2)}y_3, \\ &\cdots \\ z_{M_2} &= \nu_{02}^{(M_2)} + \nu_{12}^{(M_2)}y_{F_1-1} + \nu_{22}^{(M_2)}y_{F_1}, \end{aligned} \tag{2.11}$$

where z_j and $\nu_{i2}^{(j)}$, $j = 1, 2, \cdots, M_2$; $i = 0, 1, 2$ are the estimated outputs and corresponding coefficients of the functions. In the same way, assume that F_2 functions are passed on to the third layer; this means that there are M_3 $(= C_{F_2}^2)$ partial functions generated in this layer.

$$\begin{aligned} v_1 &= \nu_{03}^{(1)} + \nu_{13}^{(1)}z_1 + \nu_{23}^{(1)}z_2, \\ v_2 &= \nu_{03}^{(2)} + \nu_{13}^{(2)}z_1 + \nu_{23}^{(2)}z_3, \\ &\cdots \\ v_{M_3} &= \nu_{03}^{(M_3)} + \nu_{13}^{(M_3)}z_{F_2-1} + \nu_{23}^{(M_3)}z_{F_2}, \end{aligned} \tag{2.12}$$

where v_j and $\nu_{i3}^{(j)}$, $j = 1, 2, \cdots, M_3$; $i = 0, 1, 2$ are the estimated outputs and corresponding coefficients of the functions. The process is repeated by imposing threshold levels of $m \geq F_1 \geq F_2 \geq F_3 \geq \cdots \geq F_l$ so that finally an unique function is selected at one of the layers. The multilayer network structure with five input arguments and five selected nodes

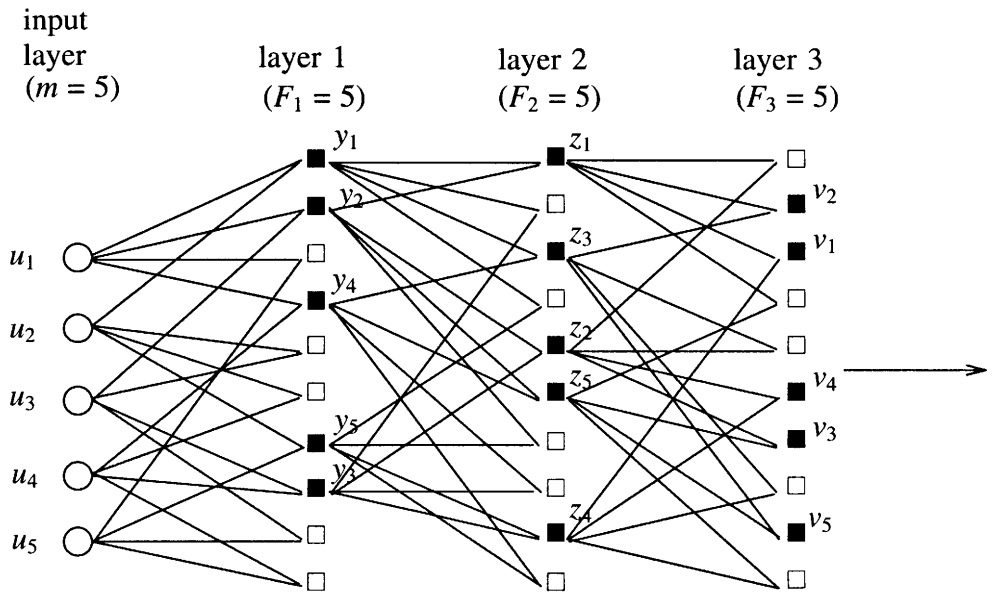

Figure 2.1. Multilayer network structure with five input arguments and selected nodes

at each layer is exhibited in Figure 2.1. For example, if the function $v_2 = \nu_{03}^{(2)} + \nu_{13}^{(2)} z_1 + \nu_{23}^{(2)} z_3$ in Equation 2.12 achieves the global minimum, then it traces back to the preceding layers to recollect the functional values of all connecting weights from the associated units. Finally, we get the optimal function in terms of the input arguments as shown below:

$$\begin{aligned} v_2 &= f(z_1, z_3) \\ &\equiv f(\, f(y_1, y_2), f(y_1, y_4)\,) \\ &\equiv f(u_1, u_2, u_3, u_5) = f(X). \end{aligned} \tag{2.13}$$

This is demonstrated in Figure 2.2. One could obtain more functions which are nearer to global measure for further evluation.

The computational aspects that are considered as the multilayer network procedure is more repetitive in nature. It is important to consider the algorithm in modules and facilitate repetitive characteristics. This is given in the last chapter of the book.

2.2 Combinatorial algorithm

This uses a single-layered structure. Summation functions are generated for all combinations of input variables; i.e., this is like "all types of regressions" in the regression analysis. Let us describe the mathematical description of a system as shown below with three input arguments.

$$y = a_0 + a_1 u_1 + a_2 u_2 + a_3 u_3, \tag{2.14}$$

where y is the estimated output, u_1, u_2, and u_3 are the input arguments, and a_i are the weights. The algorithm uses a single-layered structure because of its complexity in model building. This is given below.

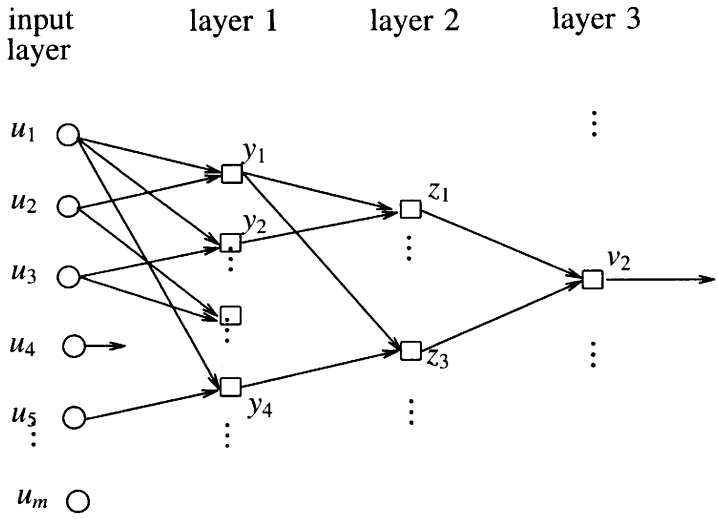

Figure 2.2. Schematic output flow to unit 2 of layer 3 in the multilayer structure

Layer level

1. Summation functions for all combinations of the input arguments u_1, u_2, u_3 (in this case there are seven units, Figure 2.3) are generated:

$$
\begin{aligned}
y_1 &= a_0^{(1)} + a_1^{(1)} u_1, \\
y_2 &= a_0^{(2)} + a_2^{(2)} u_2, \\
y_3 &= a_0^{(3)} + a_1^{(3)} u_1 + a_2^{(3)} u_2, \\
y_4 &= a_0^{(4)} + a_3^{(4)} u_3, \\
y_5 &= a_0^{(5)} + a_1^{(5)} u_1 + a_3^{(5)} u_3, \\
y_6 &= a_0^{(6)} + a_2^{(6)} u_2 + a_3^{(6)} u_3, \\
y_7 &= a_0^{(7)} + a_1^{(7)} u_1 + a_2^{(7)} u_2 + a_3^{(7)} u_3,
\end{aligned} \tag{2.15}
$$

   where y_k is the estimated output of kth unit, $k = 1, 2, \cdots, 7$; and $a_i^{(k)}$, $i = 0, 1, 2, 3$ are their connecting weights.
2. The weights are estimated by using the least squares technique with a training set at each unit.
3. Then the unit errors are compared as per the threshold objective function using a testing set, and
4. Units with selected output responses are made "on" and evaluated further.

The schematic flow of the algorithm is given in Figure 2.4.

Usage of "structure of functions." If there is increase in the input arguments, there is corresponding increase in the combinations of them. Suppose there are m variables, then the total combinations are $M_1 = (2^m - 1)$. This is the main difference of this single-layer algorithm in comparison with the multilayered algorithm described in the previous section. There is restriction on the number of input variables with which to use this algorithm as per the capacity of the computer. Efforts are given to build up the algorithm in a more

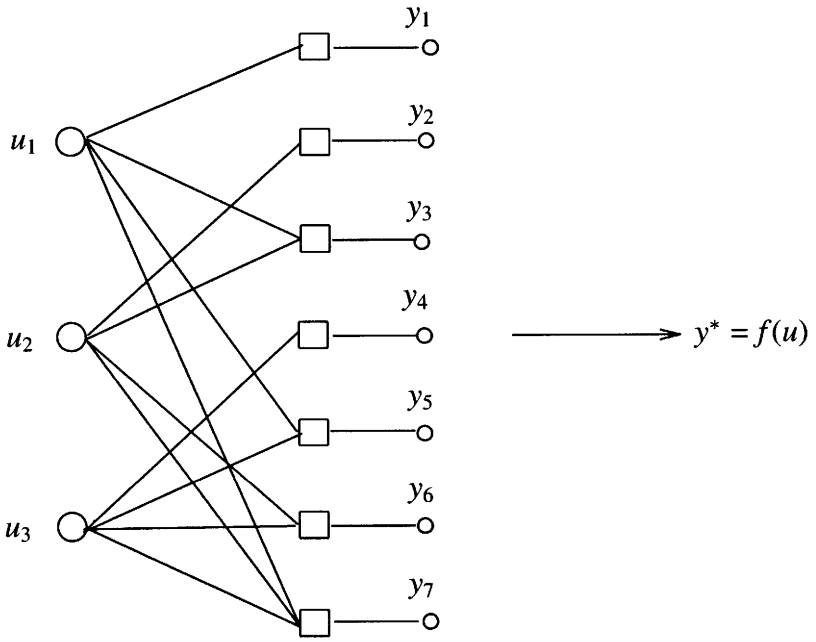

Figure 2.3. Single-layer layout of combinatorial structure

economical way in generating the functions with all combinations. It uses a binary matrix structure of "0"s and "1"s, where each row indicates a partial function with its parameters represented by "1", number of rows indicates total number of units in the layer, and number of columns indicates total number of parameters in the full description. Terms in the matrix are made equal to "0" if the parameters are not present in the function. This is called "structure of functions" (2.16), and includes the full description and the function with all arguments. Usually the "structure of functions" contains the constant term a_0 which is present in all functions:

$$\begin{array}{cccccc} i & a_0^{(i)} & a_3^{(i)} & a_2^{(i)} & a_1^{(i)} \\ 1 & 1 & 0 & 0 & 1 \\ 2 & 1 & 0 & 1 & 0 \\ 3 & 1 & 0 & 1 & 1 \\ 4 & 1 & 1 & 0 & 0 \\ 5 & 1 & 1 & 0 & 1 \\ 6 & 1 & 1 & 1 & 0 \\ 7 & 1 & 1 & 1 & 1. \end{array} \tag{2.16}$$

This is referred further in forming the normal equations for each function. Connection weights of each unit are estimated using a training set and evaluated for its threshold measure in comparison with a testing set. Finally it gives out the selected output responses as per the threshold values.

Gradual increase of complexity. As we have seen above, the combinatorial algorithm is based on an inductive approach; this is done by sorting of all possible models from a given basis of a reference function with fixed input variables. The best of them are chosen according to the external criteria. The complexity of the models is increased by sorting, which is done by gradual increase of arguments in the polynomials or partial functions. The

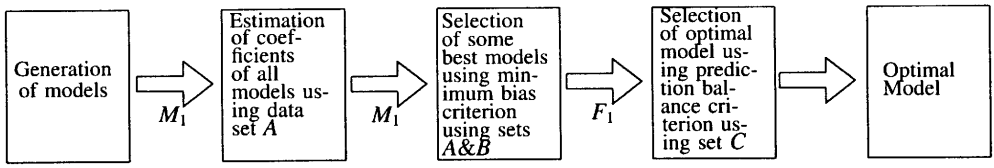

Figure 2.4. Schematic flow of single-layer combinatorial algorithm

important thing here is that no possible variants of the model that appears with the complete set are missed. Let us see how this is realized with three variables of the complete quadratic polynomial that has the form

$$y = a_0 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_1^2 + a_5x_2^2 + a_6x_3^2 + a_7x_1x_2 + a_8x_1x_3 + a_9x_2x_3. \tag{2.17}$$

There are ten terms in the full polynomial that includes the constant term a_0 ($m = 10$). Sometimes arguments like $1/x_1, 1/x_2$, and $1/x_3$ and other higher order nonlinear terms are to be included based on the global minimum attained on the external criterion. The partial polynomials are linear in coefficients a, and the least squares technique is used to estimate the coefficients by reindexing the nonlinear terms in linear form. Here we give the scheme for gradual increase of complexity in the partial functions. The scheme is as follows.

In the first step, all models with single arguments are determined.

$$y_1 = a_0, \quad y_2 = a_1x_1, \quad y_3 = a_2x_2, \cdots, \quad y_{10} = a_9x_2x_3. \tag{2.18}$$

That means there are $C_{10}^1 = 10$ partial models. Then in the second step, all models with two arguments are determined.

$$\begin{aligned} y_{11} &= a_0 + a_1x_1, \quad y_{12} = a_0 + a_2x_2, \cdots, y_i = a_0 + a_9x_2x_3 \\ y_j &= a_1x_1 + a_2x_2, \quad y_k = a_1x_1 + a_3x_3, \cdots, y_{45} = a_8x_1x_3 + a_9x_2x_3. \end{aligned} \tag{2.19}$$

There are $C_{10}^2 = 45$ partial models. Similarly, in the third step models with three arguments are built up, in the fourth step with four arguments, and so on until $C_{10}^{10} = 1$ model, which is the complete polynomial. The total number of all possible models constructed for m arguments is

$$M_1 = \sum_{s=1}^{m} C_m^s = 2^m - 1. \tag{2.20}$$

The value of M_1 increases with the increase of m; for example, if $m = 10$, then $M_1 = 1,023$ and if $m = 15$, then $M_1 = 32,767$. This algorithm with the given program at the end, where the complexity of the partial models is not changed sequentially but rather according to the binary matrix of m-digit counter, sorts all possible models for $m \leq 18$ in an acceptable time. However, the inclusion of an additional argument in the input set doubles the computational time. We give here one of the optimal sorting schemes [115] that enables us to increase the input set to $m = 23$.

2.3 Recursive scheme for faster combinatorial sorting

Suppose we are given N measurement points of output y and input variables $\mathbf{x}$ of a system. For the given output y we set up the measurement function

$$\mathbf{y} = X\mathbf{a}, \quad \text{where } \mathbf{y}[N \times 1],\ X[N \times m], \text{ and } \mathbf{a}[m \times 1]. \tag{2.21}$$

The coefficients of the model are determined by the least-squares method

$$\hat{\mathbf{a}} = (X^T X)^{-1} X^T \mathbf{y}. \tag{2.22}$$

Let us consider the question of obtaining inverse matrices of the type $(X^T X)^{-1}$ for each partial polynomial in the combinatorial algorithm. First of all we consider for one individual output; this means that at a certain stage of the algorithm we have obtained the estimates of the parameters of a partial model containing k arguments $x_1, x_2, \cdots, x_k$. Then we know that

$$\hat{\mathbf{a}}_\mathbf{k} = H_k^{-1} \mathbf{g}_k, \tag{2.23}$$

where

$$H_k = X_k^T X_k, \quad \text{and} \quad \mathbf{g}_k = X_k^T \mathbf{y} \tag{2.24}$$

denote the elements of the matrix of the normal system

$$[H_k \mid \mathbf{g}_k] \triangleq [X_k^T X_k \mid X_k^T \mathbf{y}]. \tag{2.25}$$

In estimating the coefficients $\mathbf{a}_k$, the inverse of the matrix H_k is computed by using the Gauss method. As a further step, for example, the system considers another partial model with an additional argument x_{k+1}, and the matrix of the normal system takes the form

$$[H_{k+1} \mid \mathbf{g}_{k+1}] = \left[\begin{array}{c|c|c} H_k & \mathbf{h}_{k+1} & \mathbf{g}_k \\ \hline \mathbf{h}_{k+1}^T & \vartheta_{k+1} & \gamma_{k+1} \end{array}\right], \tag{2.26}$$

where $\mathbf{h}_{k+1}[k \times 1]$ is a k dimensional vector; ϑ_{k+1} and γ_{k+1} are scalars whose values are computed using the measurements of the $(k+1)$st argument. In the combinatorial algorithm, the estimate of the vector $\hat{\mathbf{a}}_{k+1}$ is found by the least squares method; that means that one performs operations analogous to a new inversion of the matrix H_{k+1} for finding the parameter estimates $\hat{\mathbf{a}}_{k+1} = H_{k+1}^{-1} \mathbf{g}_{k+1}$. This estimate can be obtained in terms of the known H_k^{-1} and other elements $\mathbf{h}_{k+1}$ and ϑ_{k+1} as below.

$$\hat{\mathbf{a}}_{k+1} = \left[\begin{array}{c} \hat{\mathbf{a}}_k^* \\ \hline \hat{\alpha}_{k+1} \end{array}\right] = \left[\begin{array}{c|c} H_k & \mathbf{h}_{k+1} \\ \hline \mathbf{h}_{k+1}^T & \vartheta_{k+1} \end{array}\right]^{-1} \left[\begin{array}{c} \mathbf{g}_k \\ \hline \gamma_{k+1} \end{array}\right]. \tag{2.27}$$

Specifically, if we write the inverse of the matrix as

$$H_{k+1}^{-1} \triangleq B_{k+1} = \left[\begin{array}{c|c} B_k & \mathbf{b}_{k+1} \\ \hline \mathbf{b}_{k+1}^T & \beta_{k+1} \end{array}\right] \tag{2.28}$$

and solve the equation $B_{k+1} H_{k+1} = I_{k+1}$, where I_{k+1} is the identity matrix, we obtain the following formulae by inverting the above block matrices.

$$\begin{aligned} \beta_{k+1} &= 1/(\vartheta_{k+1} - \mathbf{h}_{k+1}^T \mathbf{c}_{k+1}), \quad \mathbf{c}_{k+1} \triangleq H_k^{-1} \mathbf{h}_{k+1}, \quad H_0^{-1} \triangleq 0. \\ \mathbf{b}_{k+1} &= -\beta_{k+1} \mathbf{c}_{k+1}, \quad B_k = H_k^{-1} + \beta_{k+1} \mathbf{c}_{k+1} \mathbf{c}_{k+1}^T. \end{aligned} \tag{2.29}$$

Substituting the results above, we get

$$\begin{aligned} \hat{\mathbf{a}}_k^* &= H_k^{-1} \mathbf{g}_k + \beta_{k+1} \mathbf{c}_{k+1} (\mathbf{c}_{k+1}^T \mathbf{g}_k - \gamma_{k+1}) \\ \hat{\alpha}_{k+1} &= -\beta_{k+1} (\mathbf{c}_{k+1}^T \mathbf{g}_k - \gamma_{k+1}). \end{aligned} \tag{2.30}$$

Thus, the new estimate $\hat{\mathbf{a}}_{k+1}$ can be expressed explicitly in terms of the estimate already calculated.

$$\hat{\mathbf{a}}_{k+1} = \begin{bmatrix} \hat{\mathbf{a}}_k^* \\ - - - \\ \hat{\alpha}_{k+1} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{a}}_k - \hat{\alpha}_{k+1}\mathbf{c}_{k+1} \\ - - - - - - \\ \hat{\alpha}_{k+1} \end{bmatrix}, \quad \hat{\mathbf{a}}_0 \triangleq 0. \tag{2.31}$$

It is obvious that the above operation considerably decreases the number of computations needed for estimating $\mathbf{a}_{k+1}$ in comparison with the direct determination of the matrix H_{k+1}^{-1}. If the next term x_{k+2} is adjoined to the model as another partial polynomial, then the new inverse matrix is obtained directly as

$$H_{k+1}^{-1} = \left[\begin{array}{c|c} H_k^{-1} + \beta_{k+1}\mathbf{c}_{k+1}\mathbf{c}_{k+1}^T & -\beta_{k+1}\mathbf{c}_{k+1} \\ - - - - - - - - - - & - - - - - - \\ -\beta_{k+1}\mathbf{c}_{k+1}^T & \beta_{k+1} \end{array}\right]. \tag{2.32}$$

The above recursive technique is convenient to use in constructing models of the partial polynomials of gradually increasing complexity that begin with a single argument. This type of approach is called "method of bordering."

The above table of "structure of functions" (Equation 2.16) shows the sequential change in the states of the binary counter that corresponds to the changes in structures of partial polynomials. In the shift from $1 \rightarrow 3$ or $2 \rightarrow 3$, $4 \rightarrow 5$ and 6, and $5 \rightarrow 7$ or $6 \rightarrow 7$ models, one can notice that a new term is added; the inverse matrices in these cases can simply be computed using the above recursive algorithm. This type of combinatorial scanning of models accelerates the calculation of the model coefficients and reduces the computational time; one has to think about the optimum way of utilizing the computer memory to store these inverse matrices as the number of arguments increases. The solution is to have an optimum way of sequencing the rows of binary matrix in a specific way. For example, let us consider the sequencing as follows:

i	$a_0^{(i)}$	$a_3^{(i)}$	$a_2^{(i)}$	$a_1^{(i)}$
1	1	0	0	1
2	1	0	1	1
3	1	0	1	0
4	1	1	1	0
5	1	1	1	1
6	1	1	0	1
7	1	1	0	0.

(2.33)

In this sequencing, one can notice that from $1 \rightarrow 2$ one argument is added, from $2 \rightarrow 3$ one argument is eliminated, and so on. When an argument is added, the above recursive procedure can be used for obtaining the new inverse matrix. When an argument is eliminated, a new procedure which works in reverse is needed; one can introduce the inverse operation of it to calculate H_k^{-1} and $\mathbf{a}_k$ from the known H_{k+1}^{-1} and $\hat{\mathbf{a}}_{k+1}$ as

$$H_{k+1}^{-1} = \left[\begin{array}{c|c} B_k & \mathbf{b}_{k+1} \\ - - - & - - - \\ \mathbf{b}_{k+1}^T & \beta_{k+1} \end{array}\right], \quad \hat{\mathbf{a}}_{k+1} = \begin{bmatrix} \hat{\mathbf{a}}_k^* \\ - - - \\ \hat{\alpha}_{k+1} \end{bmatrix}, \tag{2.34}$$

and eliminate the $(k+1)$st argument by

$$\begin{aligned} H_k^{-1} &= B_k - \frac{1}{\beta_{k+1}}\mathbf{b}_{k+1}\mathbf{b}_{k+1}^T, \\ \hat{\mathbf{a}}_k &= \hat{\mathbf{a}}_k^* - \frac{\hat{\alpha}_{k+1}}{\beta_{k+1}}\mathbf{b}_{k+1}. \end{aligned} \tag{2.35}$$

This type of refinement of the recursive algorithm yields maximum decrease in the time of total scanning of the models at the cost of increased memory utilization. In the above "structure of functions" (Equation 2.33), one argument is added to the models in the shift from $1 \rightarrow 2$, $3 \rightarrow 4$, and $4 \rightarrow 5$; and in the shift from $2 \rightarrow 3$, $5 \rightarrow 6$, and $6 \rightarrow 7$, one argument is eliminated. The above procedure with the alternate use of recursive and inverse recursive routines can be used as required for computing H_k^{-1} and H_{k+1}^{-1} matrices alternately from an initial matrix H_k^{-1}. The least square error ε_k^2 can be calculated after estimating the coefficients $\hat{\mathbf{a}}_k$.

$$\varepsilon_k^2 = (\mathbf{y}_A - \hat{\mathbf{y}}_A)^T(\mathbf{y}_A - \hat{\mathbf{y}}_A) \equiv \mathbf{y}_A^T\mathbf{y}_A - \hat{\mathbf{a}}_k^T\mathbf{g}_A, \tag{2.36}$$

where $\mathbf{g}_A = X_A^T\mathbf{y}_A$; here the subscript A corresponds to the training set. The recurrent algorithm enables us to compute the least squares error ε_{k+1}^2 recursively when a new argument is added.

$$\varepsilon_{k+1}^2 = \varepsilon_k^2 + \hat{\alpha}_{k+1}(\gamma_{k+1} - \mathbf{c}_{k+1}^T\mathbf{g}_k) \equiv \varepsilon_k^2 + \frac{\hat{\alpha}_{k+1}^2}{\beta_{k+1}}. \tag{2.37}$$

2.4 Multilayered structures using combinatorial setup

One version of a multilayered structure is that the combinatorial algorithm could be realized at each layer of the multilayer network structure by keeping the limit on the "freedom of choice" at each layer. The unit outputs are fed forward layer by layer as per the threshold measure to obtain the global output response for optimal complexity. This structure is exhibited in Figure 2.5 with three input arguments and three selected nodes at each layer.

2.5 Selectional-combinatorial multilayer algorithm

Here is another version of a multilayered structure that is called a selectional-combinatorial algorithm that realizes the above recursive procedure [116] [49]. The general outline of the algorithm is as follows.

In the first layer, all models containing single arguments are estimated, and some of the best are selected as per certain external criteria and passed on to the next layer. In the second layer, different arguments are selected and added to these models, which improved the response as per the external criteria. This continues until it deteriorates. In contrast to the original multilayer set up, it does not pass on the outputs of the units. The multilayer error is not passed on because of its retainment of the original basis functions; their number of arguments coincides with the layer number, and the total number of layers cannot exceed m.

The important aspect of this algorithm is its realization of the recursive procedure for successive estimation of coefficients of the partial models according to the least squares method. The matrix of normal equations for a model is represented with $l+1$ arguments that is obtained from the complete normal matrix $[m \times (m+1)]$ matrix $[H \mid \mathbf{g}] = [X^TX \mid X^T\mathbf{y}]$ using the form

$$H_{l+1} = \left[\begin{array}{c|c} H_l & \mathbf{h}_{l+1} \\ \hline \mathbf{h}_{l+1}^T & \vartheta_{l+1} \end{array}\right], \quad \mathbf{g}_{l+1} = \left[\begin{array}{c} \mathbf{g}_l \\ \hline \gamma_{l+1} \end{array}\right]. \tag{2.38}$$

The coefficients $\hat{\mathbf{a}}_{l+1}$ are estimated using the following:

$$\beta_{l+1} = 1/(\vartheta_{l+1} - \mathbf{h}_{l+1}^T\mathbf{c}_{l+1}), \quad \mathbf{c}_{l+1} = H_l^{-1}\mathbf{h}_{l+1},$$

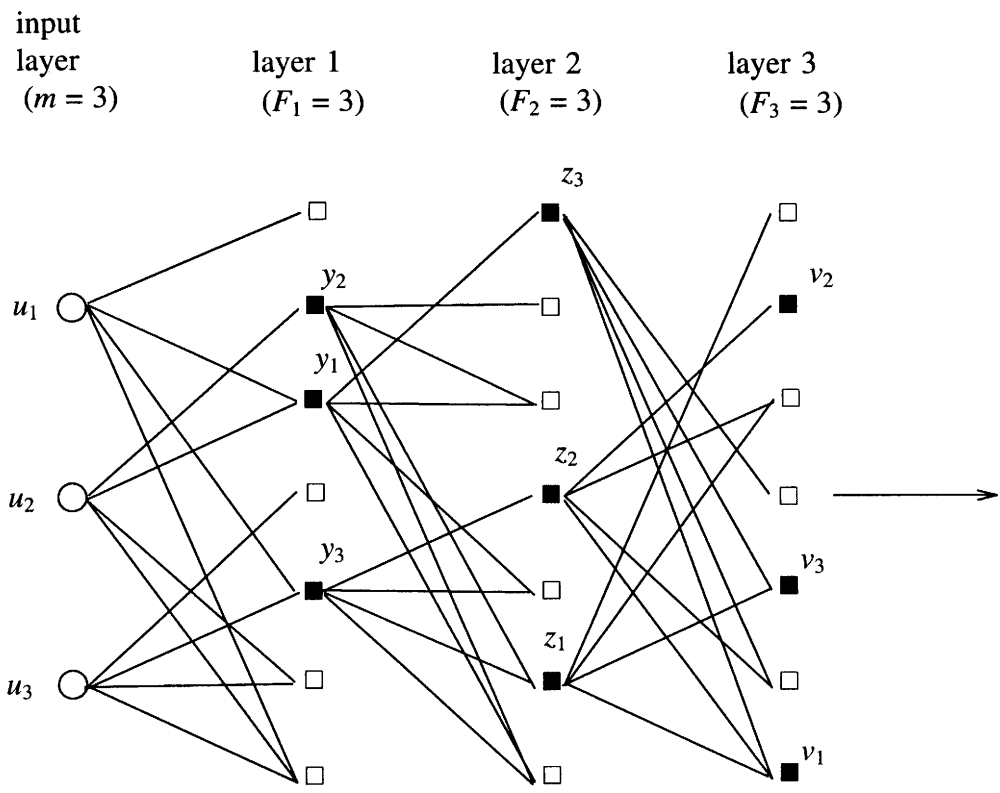

Figure 2.5. Multilayer structure with restricted combinatorial set up at each layer

$$\hat{\mathbf{a}}_{l+1} = \begin{bmatrix} \hat{\mathbf{a}}_l \\ - - - \\ \hat{\alpha}_{l+1} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{a}}_l - \hat{\alpha}_{l+1}\mathbf{c}_{l+1} \\ - - - - - - - \\ \beta_{l+1}(\gamma_{l+1} - \mathbf{g}_l\mathbf{c}_{l+1}) \end{bmatrix}. \tag{2.39}$$

Let us explain the sequence of calculations of the $(l+1)$st layer from the lth layer.

1. For each ith model ($i = 1, \cdots, M_1$) of the first layer ($l = 1$), the inverse matrices $H_1^{-1}(i)$ are calculated. These matrices will be updated at all the consecutive layers using the recursive procedure. Let us assume that $H_l^{-1}(i)$ are the inverse matrices at the lth layer and F_l models are selected as per the external criteria.
2. Partial models of the $(l+1)$st layer are generated by an ordered addition to each selected ith model of one of the arguments absent from it that correspond to the zeroth elements of the binary structure of the vector for ith model. Here the partial models M_{l+1} are generated with the complexity $l+1$ arguments. Each new model is uniquely defined from the preceding F_l models of lth layer, and the x_j is the new added argument.
3. Estimates $\hat{\mathbf{a}}_{l+1}(i,j)$ are calculated for all M_{l+1} models using the recursive procedure for each i on the basis of the matrix $H_l^{-1}(i)$. From the denominator terms, one can easily sort out even the ill-conditioned normal matrix.
4. The values of the external criteria are computed for each model.
5. The best F_{l+1} models are chosen for the next $(l+2)$nd layer from the condition of improvement of the minimal value of the external criterion; for example, $\Delta(B)$ is

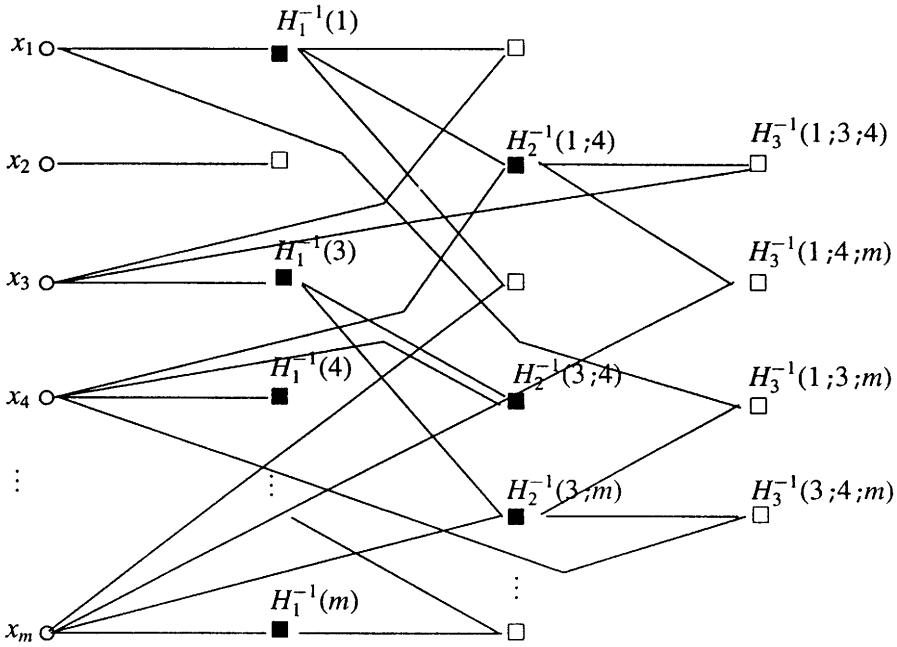

Figure 2.6. Multilayered version of selectional-combinatorial algorithm; the inverse matrices $H_l^{-1}(i)$ are updated layer after layer using the recursive technique

obtained at the preceding layer l.

$$F_{l+1} = \sum_{p=1}^{M_{l+1}} \delta_p, \qquad \delta_p = \begin{cases} 1 & \text{if } \Delta_p(B) < \theta_l \\ 0 & \text{if } \Delta_p(B) \geq \theta_l, \end{cases} \tag{2.40}$$

$$\text{where } \theta_l = \min_{i \in F_l} \Delta_i(B).$$

This procedure is applicable in a strict sense when the external criterion actually behaves like an ideal one in selecting a unimodel.

In general, one can choose selection of models on a competitive basis as in the usual multilayer algorithm.

To overcome the possible local minimum when $l+1 \ll m$ and $F_{l+1} = 0$, it is better to fix the lower boundary (for example, $F_{l+1_{min}} = m - (l+1)$) and an upper boundary F_{max}. The freedom of choice at the $(l+1)$st layer is determined with the constraint

$$F_{l+1_{min}} \leq F_{l+1} \leq F_{max}. \tag{2.41}$$

6. When $F_{l+1} = 0$, the procedure is stopped automatically which would indicate the minimum is achieved at the previous layer.

The schematic diagram of this algorithm is shown in Figure 2.6 with the passage of inverse matrix H and an additional term from layer to layer.

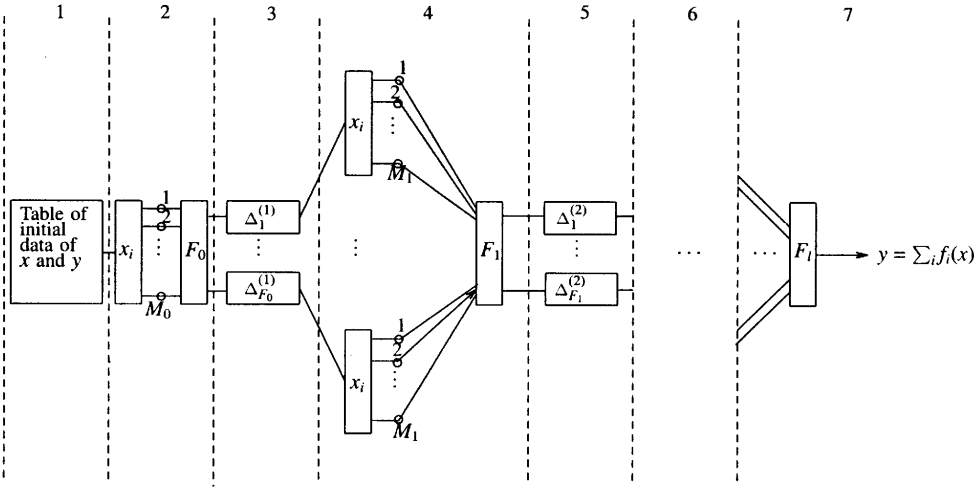

Figure 2.7. Schematic illustration of front propagation algorithm with the calculation of output error residuals, where $M_0, M_1, \cdots$ denote the total partial models; $F_0, F_1, \cdots, F_l$ are the number of best models (freedom of choice) at zeroth, first and last layers correspondingly

2.6 Multilayer algorithm with propagating residuals (front propagation algorithm)

This algorithm is built up based on forwarding the output errors to the next layers as outputs and using the combinatorial induction on original input variables at each layer. The schematic flow of such algorithm is given in Figure 2.7.

Each block of the figure is explained below.

1. Table of initial empirical data points of N, m input variables of x_i and output variable y.
2. The initial layer which is called the zeroth layer uses the combinatorial algorithm in choosing the best F_0 models.
3. The first differences (error residuals for each model) which are denoted as $\Delta_i^{(1)}, i = 1, \cdots, F_0$ are computed. Each vector, $\Delta_i^{(1)}$ is of $[N \times 1]$; $\Delta_i^{(1)} = \mathbf{y} - \hat{\mathbf{y}}_{(i)}$, where $\hat{\mathbf{y}}_{(i)}$ is the vector of estimated output corresponding to ith model.
4. This is the first selection layer in which the F_0 vectors of first differences are used as output variables independently at each combinatorial sorting procedure. The best F_1 models are chosen.
5. The second differences $\Delta_j^{(2)}, j = 1, \cdots, F_1$ are computed; $\Delta_j^{(2)} = \Delta^{(1)} - \hat{\Delta}^{(1)}$, where $\Delta^{(1)}$ is the vector of actual values of first differences from the first layer and $\hat{\Delta}^{(1)}$ is the vector of estimated output corresponding to each jth model.
6. This block denotes the similar follow-up of layers and calculation of further residuals.
7. Last layer with the model of optimal complexity.

Some features of the algorithm

One can see below the overall effect of the forward propagation of residuals.

First-differences (residuals) at layer "0" are computed as

$$\Delta^{(1)} = \mathbf{y} - \hat{\mathbf{y}}$$

$$\equiv \mathbf{y} - X\hat{\mathbf{a}}^{(0)}, \tag{2.42}$$

where $\mathbf{y}, \hat{\mathbf{y}}$ and $\Delta^{(1)}$ are the actual, estimated and first difference vectors with the dimension of $[N \times 1]$, $\hat{\mathbf{a}}^{(0)}$ is the vector of least squares estimates of coefficients with the dimension $[m \times 1]$, and matrix X is the initial data of $[N \times m]$.

Second-differences (residuals) at layer "1" are computed as

$$\begin{aligned} \Delta^{(2)} &= \Delta^{(1)} - \hat{\Delta}^{(1)} \\ &\equiv \mathbf{y} - X\hat{\mathbf{a}}^{(0)} - X\hat{\mathbf{a}}^{(1)} \\ &= \mathbf{y} - X(\hat{\mathbf{a}}^{(0)} + \hat{\mathbf{a}}^{(1)}), \end{aligned} \tag{2.43}$$

where $\hat{\mathbf{a}}^{(1)}$ is the vector of estimated coefficients at the first layer.

The propagating residuals help in the finer adjustment of the coefficients as the process proceeds layer by layer and ultimately an optimal model is obtained.

The external criterion $c3^2 \triangleq \bar{\eta}_{bs}^2 + \bar{\Delta}^2(C)$ decreases monotonically with the prolifeiration of the selection layers and a relatively small decrease in it serves as a signal to stop the procedure. In comparison with the original multilayer algorithm, the objective nature of the choice of model is not fully conserved because the error decreases monotonically.

This algorithm is worked out for finite-difference reference functions [54]. In the absence of constraints on the coefficients, this algorithm with forward error propagations resembles the exponential-harmonic algorithm.

2.7 Harmonic Algorithm

The principal aim of this algorithm is to extend the use of the inductive self-organization principle to identify the oscillatory processes. It is assumed that the effective reference functions of such processes are in the form of the sum of harmonics with nonmultiple frequencies. The harmonic function is composed of several sinusoids with arbitrary frequencies which are not necessarily related. This type of function produces a multifrequency resultant and exhibits similar spectral characteristics. A balance relation plays an important role in obtaining the frequencies of the process and as an objective function in selecting the optimal trend in the multilayered structure. The algorithm is explained below with the derivation of the balance relation.

Suppose a function $f(t)$ is a process having the sum of m harmonic components with pairwise discrete frequencies $w_1, w_2, \cdots, w_m$.

$$\begin{aligned} f(t) &= \sum_{k=1}^{m} [\mathcal{A}_k \sin(w_k t) + \mathcal{B}_k \cos(w_k t)] \\ &\equiv \sum_{k=1}^{m} \Phi(w_k, t), \end{aligned} \tag{2.44}$$

where $\mathcal{A}_k$ and $\mathcal{B}_k$ are the coefficients; and $w_i \neq w_j$, $i \neq j$, $0 < w_i < \pi$, $i = 1, 2, \cdots, m$. The process has a total interval length of N $(1 \leq t \leq N)$ and points at discrete intervals of time δ_t.

For a fixed point i and any p of Equation 2.44, one can obtain the formula:

$$f(i+p) + f(i-p) = 2 \sum_{k=1}^{m} \cos(pw_k)\ \Phi(w_k, i). \tag{2.45}$$

Summing Equation 2.45 for p from 0 to $m-1$, with weighing coefficients $\mu_0, \mu_1, \cdots, \mu_{m-1}$, we derive a relation:

$$\begin{aligned}\sum_{p=0}^{m-1} \mu_p [f(i+p)+f(i-p)] &= 2\sum_{k=1}^{m} \Phi(w_k, i)[\mu_0 + \sum_{p=1}^{m-1} \mu_p cos\,(pw_k)] \\ &= 2\sum_{k=1}^{m} \Phi(w_k, i)\ cos\,(mw_k) \\ &= f(i+m)+f(i-m). \end{aligned} \tag{2.46}$$

This is considered as a balance relation of the process.

$$b_i = [f(i+m)+f(i-m)] - \sum_{p=0}^{m-1} \mu_p [f(i+p)+f(i-p)]. \tag{2.47}$$

If the process is expressed exactly in terms of a given sum of harmonic components, then $b_i = 0$; i.e., the discrete values of $f(t)$, which are symmetric with respect to a point $i\,(m+1 \leq i \leq N-m)$, satisfy the balance relation. The coefficients μ_p are independent of i.

The trigonometric relation used in the calculations of Equation 2.46 is:

$$\mu_0 + \sum_{k=1}^{m-1} \mu_p\, cos\,(pw_k) = cos\,(mw_k). \tag{2.48}$$

This could be formed as mth degree algebraic equation in $cos\,w$ by using the recursive trigonometric relations

$$\mathcal{D}_m(cos\,w)^m + \mathcal{D}_{m-1}(cos\,w)^{m-1} + \cdots + \mathcal{D}_1(cos\,w) + \mathcal{D}_0 = 0, \tag{2.49}$$

where $\mathcal{D}_i,\ i = 0, 1, \cdots, m$ are the functions of μ_p.

It is possible to determine uniquely the coefficients $\mu_p,\ p = 0, 1, \cdots, m-1$ from the relation (Equation 2.46) for $i = m+1, \cdots, N-m$. $(N-m)-(m+1) \geq m-1$; i.e., $N \geq 3m$. Substituting the values of μ_p in Equation 2.49, it can be solved for m frequencies w_k of harmonics by using standard numerical techniques. These frequencies are passed through a multilayered network to form various combinations of the harmonic components. The parameters of the harmonic components $\mathcal{A}_k$ and $\mathcal{B}_k$ are estimated at each unit by using the least squares technique. The optimal trend is obtained at one of the units by considering the balance relation as a threshold objective function.

Suppose $y(t)$ is the given discrete time series data and is to be identified for its harmonic trend $f(t)$. The data are to be separated into two data sets as training set points N_A and testing set points N_B. We can allot some more points N_C as a checking set for checking the trend, i.e., $N = N_A + N_B + N_C$. The maximum number of harmonics is chosen as M_{max} $(< N/3)$. The coefficients μ_p are estimated by using the least squares technique by forming the balance equations with the training set. The system of equations has the form:

$$\begin{aligned}\sum_{p=0}^{m-1} \mu_p[y(i+p)+y(i-p)] &= y(i+m)+y(i-m); \\ i &= m+1, \cdots, N_A - m. \end{aligned} \tag{2.50}$$

By substituting the values of μ_p, the frequencies of the harmonics are determined by solving the mth degree Equation 2.49. This has m roots that uniquely determine m frequencies w_k.

These frequencies are fed through the input layer of a multilayered structure as components of the harmonic terms. The procedure of complete sifting of trends would take place by multilayer selection of trends using the inductive principle. The linear normal equations are constructed in the first layer for any $1 \leq m \leq M_{max}$ number of harmonics, and the coefficients $\mathcal{A}_k$ and $\mathcal{B}_k$ are estimated for all combinations of M_{max} harmonics based on the training set using the least squares technique. All harmonic trends are evaluated for their threshold values in comparison with the testing set, and the output errors of the best trends from F_1 units are fed forward as inputs to the second layer. This procedure is repeated in all subsequent layers. The complexity of the model increases layer by layer as long as the value of the "imbalance" decreases on the testing set points N_B. The balance criterion B (Equation 2.51) is used as the objective function which takes into account the balance relation, b_i (Equation 2.47).

$$B \triangleq \sum_{i=m+1}^{N_B - m} b_i^2 \rightarrow min. \tag{2.51}$$

This unique solution is guaranteed as it is in the multilayer algorithm; the performance of the optimal trend can be tested further using the checking set N_C.

2.8 New algorithms

According to the form of the reference functions, the inductive learning algorithms can be divided into several main classes that could be constructed based on the addition (additive algorithms) or multiplication (multiplicative algorithms); they could be extended further as additive-multiplicative and multiplicative-additive with the factors considered as unit, integer, or noninteger powers. In addition to these, there are other algorithms like correlational and orthogonalized (generalized) algorithms. Let us first give some of the types of polynomials and later study some algorithms.

(i) *Additive polynomials with unit powers of the factors*:

$$y = a_0 + \sum_{j=1}^{m} a_j x_j, \tag{2.52}$$

where m is the number of independent variables; a are the coefficients; y and x are the output and input variables correspondingly.

(ii) *Multiplicative polynomials with unit powers of the factors*:

$$y = a. \prod_{j=1}^{m} x_j, \tag{2.53}$$

where a is the single coefficient; y and x are the output and input variables; and m is the number of independent variables.

(iii) *Multiplicative-additive polynomial with unit powers of factors*: This can have different forms as per the complexity of the terms. One of the forms is given as

$$y = \left\{ \sum_{k=1}^{m} a_k . \prod_{j=1}^{k} x_j \right\}, \tag{2.54}$$

and another form can have factors in integer or noninteger powers

$$y = \left\{ \sum_{k=1}^{s} a_k . \prod_{j=1}^{m} x_j^{p_{kj}} \right\}, \quad p_{kj} \in \{0, \Delta h, \cdots\}, \tag{2.55}$$

where s denotes the complexity of the equation, and Δh is the least fractional power (a certain minimum fractional power of the original factors is indicated; for example, $x_j^{0.5}$ and $x_j^{-0.5}$). The model of optimal complexity in this case can be represented in the form of linear and multiplicative terms with the powers not higher than the powers specified in advance.

Multiplicative-additive (generalized) algorithm

The algorithm described below allows one to obtain models with the multiplicative-additive terms [51].

When the basic inductive algorithms, where the variables have integer powers, do not lead to unbiased and accurate predictions, it is necessary to shift the solution space to another region of functional space; for example, to the region of polynomials with other than integer powers of generalized arguments. This is possible with the following multiplicative-additive algorithm. First of all, one has to choose certain multiplicative models with optimal complexity on the basis of the external criteria. An original model is represented in the form of a product of given arguments with unknown powers;

$$y = a_0 x_1^{k_1} x_2^{k_2} x_3^{k_3} \cdots x_m^{k_m}. \tag{2.56}$$

This can be rewritten in the following form by taking logarithms on both sides:

$$\ln y = \ln a_0 + k_1 \ln x_1 + k_2 \ln x_2 + \cdots + k_m \ln x_m. \tag{2.57}$$

Using the original data table of the quantities $y, x_1, \cdots, x_m$, a new data table for the variables with the logarithmic values can be set up. Data is separated into training, testing, and examining sets. Several partial, but best, models can be chosen by using one of the inductive learning algorithms (combinatorial or multilayer) with the combined criteria of "minimum-bias plus prediction." By inverting the logarithms of these optimal models one can obtain the best multiplicative models of the given process.

At the second level, to obtain the generalized multiplicative-additive model, we combine the selected multiplicative models into a single complete polynomial as

$$y = b_0 + b_1 \hat{y}_1 + b_2 \hat{y}_2 + \cdots + b_l \hat{y}_l + \cdots, \tag{2.58}$$

where y is the desired output of the process; $\hat{y}_j, j = 1, 2, \cdots, l, \cdots$ are the estimated outputs of the selected multiplicative models; and b are the coefficients.

The combinatorial algorithm enables us to obtain a unique optimal model. This model can be rewritten in terms of the original input variables; this is in the form of the multiplicative-additive model with the sum of covariance terms with noninteger powers of the factors and their products which could be used for further analysis.

Sometimes, at the first level, when the conditional equations are formed, it is necessary to take the logarithms of negative quantities. This situation can be avoided either by discarding that particular conditional equation or by reformulating the data in advance. The constant term is chosen in such a way that the error caused by applying the least squares technique to the logarithms of the variables, rather than to the variables themselves, is compensated.

Algorithm for correlation models

Rosenblatt [106] asserts that an infinite perceptron can execute a classification of images according to classes without any *a priori* information. Analogously, one might assert that a

model of optimal complexity can be found by sorting the points of the entire function space without any *a priori* specification (or "prompting"). Actually, infinitely large sortings are impossible either in a perceptron or by using inductive algorithms. That's why the class of functions that can be used to choose the complete description must be chosen on the basis of certain *a priori* "prompting." The majority of the inductive algorithms are based on the following "prompting": the complete description is designed as a polynomial which is in its linear form. This means that, as per this constraint, the object is described by a linear algebraic or finite-difference equation that remains unchanged at the interpolation or prediction intervals; the stability condition for these types of equations is unchanged.

In the regions of other types of functional spaces (for example, in the correlational functional space) it can be prompted by computer with the condition of correlational stability. This condition provides a way of extending the function manifold that can be used by applying correlation models of standard form [53], [68]. Here "prompting" is set up by a complete set of support functions of the correlational type,

$$K_y(\tau) = \frac{1}{T-\tau-1}\sum_{i=\tau}^{T} y_i y_{i-\tau}; \quad 1 \le i \le T, \tag{2.59}$$

where τ is the displacement along the axis.

The sum of the discrepancies of these equations from the values of the correlation function for $\tau = 1, 2, \cdots, n$ is

$$\Delta_\Sigma^2 = \Delta_1^2 + \Delta_2^2 + \cdots + \Delta_\tau^2, \tag{2.60}$$

where $\Delta_1, \Delta_2, \cdots, \Delta_n$ are the mismatchings and $\Delta_j = K_y(\tau) - K_y(\tau + j)$.

One can obtain a correlational model of standard structure for a fixed value of τ. The combinatorial algorithm is used to choose the optimal number of mismatchings of the correlation equations. For each sum of the discrepancies, a system of normal equations are formed as

$$\frac{\partial \Delta_\Sigma^2}{\partial y_1} = 0, \quad \frac{\partial \Delta_\Sigma^2}{\partial y_2} = 0, \cdots. \tag{2.61}$$

The model of optimal complexity contains the optimal set of mismatchings. Differentiating this sum with respect to the first sought prediction y_{pr} at the point T, the inverse transformation of the correlation function into a prediction that best minimizes the sum of discrepancies Δ_Σ is obtained. The obtained equation $y_{pr}^3 + ay_{pr} + b = 0$ is the correlational predicting model and the prediction is the real root of this equation. The procedure is repeated for calculating the prediction at the point $T + 1$ by replacing the interval for determination of the estimates of the correlation function.

In the *case of two-dimensional models*, we have

$$\begin{aligned}
K_y(\tau_i, \tau_j) &= \frac{1}{(N_i - \tau_i - 1)(N_j - \tau_j - 1)} \sum_{i=\tau_i+1}^{N_i} \sum_{j=\tau_j+1}^{N_j} y_{i,j} y_{i-\tau_i, j-\tau_j}; \\
K_y(\tau_i + 1, \tau_j) &= \frac{1}{(N_i - \tau_i)(N_j - \tau_j - 1)} \sum_{i=\tau_i+2}^{N_i} \sum_{j=\tau_j+1}^{N_j} y_{i,j} y_{i-\tau_i-1, j-\tau_j}, \\
&\quad (1 \le i \le N_i,\ 1 \le j \le N_j); \\
\Delta_{10} &= K_y(\tau_i, \tau_j) - K_y(\tau_i + 1, \tau_j),
\end{aligned} \tag{2.62}$$

where τ_i is the displacement along the horizontal line corresponding to i; τ_j is the displacement along the vertical line corresponding to j; N_i and N_j are the number of discrete values of the function along the horizontal and vertical lines respectively; and Δ_{10} is the discrepancy.

One has to compute all discrepancies for the specified pattern area of the two-dimensional discrete field and obtain a correlational model of standard structure from the system of normal equations. Differentiating this sum with y_{pr}, which denotes the predicted value of the variable, the inverse transformation of the correlation function is obtained as a cubic model; the real root of the function represents the value of the prediction. The usage of combinatorial algorithm is similar to the above case.

Case of multivariable (multifactor) fields. An advantage of the algorithm with correlational models over the harmonic algorithm is the possibility of solving multifactor problems. For example, if two variables (the output y and its factor x) are indicated in each cell of the pattern in the two-dimensional field, the functions similar to the above must be formed corresponding to two autocorrelation functions ($K_{(y_i,y_j)}$ and $K_{(x_i,x_j)}$) and one cross-correlation ($K_{(y_i,x_i)}$).

Differentiating the sum of squares of the discrepancies and setting $\frac{\partial \Delta_\Sigma^2}{\partial y_{pr}} = 0$, one obtains an estimate of the extrapolation that is averaged in the mean-square sense. A nonlinear equation is obtained if the terms include a discrepancy of the expression for correlation coefficient without displacement; otherwise, linear equations are obtained. The variety of multi-dimensional autocorrelation and cross-correlation functions and the set of their ordinates make it possible to obtain a corresponding variety of correlational models of standard structure. The optimal model is selected by using the combinatorial or multilayer algorithm with the help of external criteria. This is subjected to the sorting on the basis of the criteria.

Multilayer algorithm using the correlation models of standard structure. Let us assume that the initial data sample of N points is supplied for the output variable y and the input arguments (factors) $x_1, x_2, \cdots, x_m$; and that the number of data points $N(= A \cup B)$ are comparatively small, where A and B are the training and testing sets correspondingly.

(i) At the first layer, partial descriptions in the form of systems of equations are formed using the output variable y with a pair of input variables x_i and x_j;

$$
\begin{aligned}
K_{yx_i}(\tau) &= \frac{1}{N_\tau} \sum_k y_k x_{i,k-\tau}, \\
K_{yx_j}(\tau) &= \frac{1}{N_\tau} \sum_k y_k x_{j,k-\tau}, \\
K_{x_i y}(\tau) &= \frac{1}{N_\tau} \sum_k x_{i,k} y_{k-\tau}, \\
K_{x_j y}(\tau) &= \frac{1}{N_\tau} \sum_k x_{j,k} y_{k-\tau}, \\
K_{yy}(\tau) &= \frac{1}{N_\tau} \sum_k y_k y_{k-\tau}.
\end{aligned} \tag{2.63}
$$

It is possible to make C_m^2 such partial descriptions for each value of the displacement τ.

(ii) The values of the functions $K_{x_i y}(\tau)$ and $K_{yy}(\tau)$ are found by averaging the sequence of observations of length N_τ over several intervals. The values of y are taken as actual values from the data table.

(iii) The estimated values of the output variable $\hat{y}$ are computed from the correlation models. The best system of equations of the ordinates of the correlation function is chosen by minimizing the sum of squares of the discrepancies of that system as the relation $\Delta_{\Sigma}^2 = \sum_i \Delta_i^2$ is satisfied. The optimal vector $\hat{y}$ is obtained by solving the normal equations of the form $\frac{\partial \Delta_{\Sigma}^2}{\partial \hat{y}_i} = 0$. Calculation of $\hat{y}$ enables us to expand the initial data table; the number of columns of the table may be increased by F_1, where F_1 is the freedom-of-choice, $(F_1 < C_m^2)$.

The algorithm described above can be used for identification and prediction of the sequence of observations. This can be realized in a multilayered structure.

Correlational models provide an inverse transition from correlation functions to the original field or process for predicting it. Success with the correlation models is based on the following features.

1. Correlational models are to be constructed for stationary fields or processes, or for their remainders (obtained after removing their regular trend), where the condition for correlation stability holds.
2. The coefficients of the correlation models are to be estimated by minimizing the condition of the sum of their squared deviations.
3. The noncontradictory correlation models are to be chosen by using an inductive learning algorithm with the minimum-bias criterion.
4. Correlational models are nonphysical; i.e., they do not easily lend themselves to interpretation; that's why, when they are sorted out, one has to consider a great variety of candidate models so that the criterion indicated must be ensured a definite confidence level.

Generalized algorithm with orthogonal partial descriptions

Modeling of complex systems is frequently hindered by possible selection of initial independent variables; sometimes this might result in the loss of stability of the model structure. When one changes either the selection criterion or the composition of the data sequences of training and testing, the composition of the original arguments in a model and its structure begin to change strongly. There usually appear many models of similar quality. The stability is lost in estimating the coefficients because of the mutual dependence of the arguments, causing a mutual dependence on the corresponding coefficients. These disadvantages are especially evident in standard regression analysis where a single sequence of experimental points are used; better results are obtained by using the inductive learning algorithms. The major advantage of inductive learning algorithms is that one can avoid such biased results in identification of the object by using the minimum-bias criterion on the specific selection of the experimental points. However, the adaptation of coefficients and estimation of their confidence intervals over the entire data sample become impossible if the final form of the model contains dependent initial variables. The orthogonalized inductive algorithms allow one to improve stability in determination of coefficients. The algorithm with orthogonalized partial descriptions [102] [123] enables one to improve the stability in determining the coefficients by facilitating and (i) to use dependent variables in the set of experimental data, (ii) to obtain independent estimates of the coefficients, and (iii) to perform adaptation of the optimal model coefficients by refining their estimates over the entire sequence of the experimental points.

Let us consider that the complete polynomial of the object with dependent variables is

in the form of a Kolmogorov-Gabor polynomial:

$$y = a_0 + \sum_{j=1}^{m} a_j \prod_{i=1}^{k} x_i^{p_{ji}},$$
$$p_{ji} = 0, 1, 2, \cdots; \quad m \le l; \quad \sum_{i=1}^{k} p_{mi} \le s, \tag{2.64}$$

where y and x are the output and input variables correspondingly; s denotes the maximum allowed degree of the polynomial; l specifies the maximum allowed terms in the model; and k indicates the number of initial input variables.

Let us denote the vector forms of the data matrices;

$$Y\,[N \times 1]; \quad X = [x_{ij}], \; 1 \le i \le k, \; 1 \le j \le N, \tag{2.65}$$

where N is the number of experimental points.

Data is divided into training A and testing B sequences; $N = A \cup B$.

At the *first layer*, the actual output quantity y is projected orthogonally onto each argument from the generalized arguments $\hat{x}_{1m}$ separately for the training and testing sequences. The partial descriptions have the form

$$\hat{y}_{1m}^{A} = a_{1m}^{A}\hat{x}_{1m}, \quad a_{1m}^{A} = \frac{E(y.\hat{x}_{1m})_A}{E(\hat{x}_{1m}^2)_A},$$
$$\hat{y}_{1m}^{B} = a_{1m}^{B}\hat{x}_{1m}, \quad a_{1m}^{B} = \frac{E(y.\hat{x}_{1m})_B}{(\hat{x}_{1m}^2)_B}, \tag{2.66}$$

where E is the notation for mathematical expectation.

Each element of the set of generalized arguments is estimated from the initial variables by using the formulas

$$\hat{x}_{1m} = \prod_{i=1}^{k} x_i^{p_{mi}} - E\left(\prod_{i=1}^{k} x_i^{p_{mi}}\right),$$
$$p_{mi} = 0, 1, 2, \cdots; \quad \sum_{i=1}^{k} p_{mi} \le s. \tag{2.67}$$

The partial descriptions obtained from above are compared to each other by using the minimum-bias criterion of coefficients;

$$\eta_a^2 = \frac{(a_{1m}^{A} - a_{1m}^{B})^2}{(a_{1m}^{A})^2 + (a_{1m}^{B})^2}. \tag{2.68}$$

From this analysis, the best generalized arguments, which are denoted as $\hat{x}_1$, are selected and introduced into the model to obtain the value of a_1 that is refined (adapted) by using the entire data sample. Thus, the adaptation of coefficients is combined with the selection of variables. Then the remainder of the output quantity Δ_1 is computed and used in the subsequent layers.

$$\Delta_1 = y - a_1\hat{x}_1 \tag{2.69}$$

Subsequent selection layers are analogous to the first one. The computed remainder output quantity after the selection layer r is Δ_r;

$$\Delta_r = \Delta_{r-1} - a_r\hat{x}_r. \tag{2.70}$$

The calculations for the elements in the set of generalized arguments are also changed as follows:

$$\hat{x}_{rm} = \prod_{i=1}^{k} x_i^{p_{mi}} - E\left(\prod_{i=1}^{k} x_i^{p_{mi}}\right) - \sum_{j=1}^{r-1} b_{rm_j}\hat{x}_j,$$
$$p_{mi} = 0, 1, 2, \cdots; \quad \sum_{i=1}^{k} p_{mi} \le s. \tag{2.71}$$

The coefficients b_{rm_j} are obtained from the orthogonalized condition for $\hat{x}_{rm}$ and for each of the generalized arguments obtained during the preceding layers as

$$b_{rm_j} = \frac{E\left[\left\{\prod_{i=1}^{k} x_i^{p_{mi}} - E\left(\prod_{i=1}^{k} x_i^{p_{mi}}\right)\right\}\hat{x}_j\right]}{E\left(\hat{x}_j^2\right)}. \tag{2.72}$$

This shows that a partial description obtained from each layer is orthogonal to all previous descriptions. That is why the estimates of all coefficients a_r are independent from one another, which allows us to adapt them separately after each selection. One has to keep in mind that the orthogonalization and centering of the generalized variables are performed separately as shown above using the training and testing sets during the selection of variables, and that the refining of the model coefficients is performed on the entire sample.

In view of the linearity of the orthogonalization transformation with respect to the initial variables, one can use an inverse transformation on the final model obtained. Such an inverse transformation is necessary for interpretation of the results of the modeling.

Stopping rule. The independence of the model coefficients enables one to obtain independent estimates of their confidence intervals. The confidence interval of each coefficient is obtained as

$$d(a_r) = \sqrt{\frac{\sigma(\hat{x}_r)}{\sigma(\Delta_r)}} \cdot t_{1-\frac{\alpha(n-2)}{2}}, \tag{2.73}$$

where $d(a_r)$ is the confidence interval of coefficient a_r; $\sigma(\hat{x}_r)$ and $\sigma(\Delta_r)$ are the estimated variances of the generalized argument and the remainder quantity, correspondingly; and $t_{1-\frac{\alpha(n-2)}{2}}$ is the quantile of the student t-distribution for the probability $(1-\alpha)$ and $(n-2)$ degrees of freedom.

The student criterion can be used as a second external criterion in order to determine the optimum complexity of the model. When $\frac{|a_r|}{d(a_r)} > 1$, then the coefficient a_r is significant with a probability of $(1-\alpha)$ and is included in the model. When this condition is not satisfied, the selection is stopped. The coefficients of the final model are statistically significant, making the model highly reliable. Thus, the criterion used for stopping the selection could also be the principal criterion used in the algorithm.

It is concluded that (i) the algorithm with orthogonal partial descriptions ensures stability of the model structure and of the estimates of its coefficients for complex system modeling with changing dependent variables, and (ii) the criterion proposed for measuring the significance of coefficients enables us to obtain a statistically reliable model with optimal complexity.

The structures of inductive learning algorithms are analogous to each other for different reference functions. The three basic forms of structures, single-layer combinatorial, multilayer, and harmonic algorithms are given here. Obtaining finite-difference models with lagging terms using the former type of polynomial algorithms is analogous to obtaining a harmonic model using the later type of harmonic algorithm [49]. The different forms of multilayer structures, multilayer algorithms with forward error propagations, selectional-combinatorial with the realization of recursive procedure and with restrictions on the freedom of choice are covered. There is another way of looking at the algorithms according to the types of polynomials; various recently developed algorithms are briefly presented.

However, the properties of the algorithms with different types of reference functions depend only on the structure. Note that (a) combinatorial algorithm does not produce errors due to multilayeredness and does not admit "loss" of optimal model; (b) multilayer algorithm

without propagating errors has an explicitly expressed minimum which defines the model complexity because of an external criterion; (c) multilayer algorithm with propagation of errors has the error of monotonic nature, and the choice of model is made according to the "left rule."

Multilayer algorithms enable us to obtain polynomials with more terms than number of points in the data sample and with number of harmonics exceeding the number of harmonics of the first layer. By appropriately choosing the combined criterion (particularly choosing the minimum bias criterion as one of them), one can arrange for all terms of the original function to be reproduced using a very small number of data points. The results of self-organizing models on the basis of multilayer algorithms with and without front propagation of errors can coincide only for a rather large sample of initial data and in case of an identification problem using the minimum-bias criterion.

Multiplicative-additive algorithms of correlation models and algorithms with orthogonalized partial functions extend the region of functional space used with the inductive approach, and thus increase the possibility of solving complex problems.

3 LONG-TERM QUANTITATIVE PREDICTIONS

The subjective character of the models and the inaccuracy of long-term predictions obtained by various authors who used probabilistic and simulation methods have somewhat undermined the authority of cybernetics. The self-organization method should be able to change such a situation drastically. A computer can become an arbiter of controversies between various scientists only when objective methods are placed at its disposal. This means that we are approaching the creation of a collective man-machine superintellect that will be capable of solving the most complicated problems in prediction and control. The domain of activity includes the problems of nature that require more knowledge and skill than possessed by human experts. A computer that works on the basis of inductive learning algorithms is able to participate in the creative process as an equal partner with a human being. Let us see how it is achieved considering the following facts and characteristics.

3.1 Autocorrelation functions

Statistical prediction of random processes uses empirical data of the process (its previous history) to estimate its future values by applying the probabilistic characteristics of the process and corresponding algorithms. From the prediction point of view, one of the most important characteristics which indicates the statistical connection between the values of the process separated by some interval of time τ is the correlation function A_y:

$$A_y(t_1, t_2) = A_y(\tau) = E[\overset{\circ}{y}(t_1)\,\overset{\circ}{y}(t_1 + \tau)], \tag{2.74}$$

where $\tau = t_2 - t_1$; E denotes the expected value, and $\overset{\circ}{y}(t) = y(t)$ is an m_y-centered process (m_y is the mathematical expectation of the process).

Usually normalized correlation function is used as

$$\rho_y(\tau) = \frac{A_y(\tau)}{A_y(0)}, \tag{2.75}$$

where $A_y(0)$ is the variance of the process. One of the properties of the correlation function is that it is an even function: $A_y(\tau) = A_y(-\tau)$. In practice, when dealing with ergodic stationary processes, averaging over the set of realizations is replaced with averaging over

(to be continued)

3 COMPUTATIONAL ASPECTS OF HARMONICAL ALGORITHM

This is used mainly to identify the harmonical trend of oscillatory processes [127]. It is assumed that the effective reference functions of such processes are in the form of a sum of harmonics with nonmultiple frequencies. This means that the harmonical function is formed by several sinusoids with arbitrary frequencies which are not necessarily related.

Let us suppose that function $f(t)$ is the process having a sum of m harmonic components with distinct frequencies $w_1, w_2, \cdots, w_m$.

$$f(t) = \mathcal{A}_0 + \sum_{k=1}^{m} [\mathcal{A}_k \sin(w_k t) + \mathcal{B}_k \cos(w_k t)], \tag{8.9}$$

where $\mathcal{A}_0$ is the constant term; $\mathcal{A}_k$ and $\mathcal{B}_k$ are the coefficients; and $w_i \neq w_j,\ i \neq j,\ 0 < w_i < \pi,\ i = 1, 2, \cdots, m$. The process has discrete data points of interval length of N $(1 \leq t \leq N)$.

A balance relation is derived using the trigonometric properties for a fixed point i and any p;

$$\sum_{p=0}^{m-1} \mu_p [f(i+p)+f(i-p)] = f(i+m)+f(i-m), \tag{8.10}$$

where $\mu_0, \mu_1, \cdots, \mu_{m-1}$ are the weighing coefficients. This is considered a balance relation of the process and is used as an objective function

$$b_i = [f(i+m)+f(i-m)] - \sum_{p=0}^{m-1} \mu_p [f(i+p)+f(i-p)]. \tag{8.11}$$

If the process is expressed exactly in terms of a given sum of harmonic components, then $b_i = 0$; i.e., the discrete values of $f(t)$ which are symmetric with respect to a point $i\,(m+1 \leq i \leq N-m)$ satisfy the balance relation. The coefficients μ_p are independent of i. It is possible to determine uniquely the coefficients μ_p, $p = 0, 1, \cdots, m-1$ from the balance relation for $i = m+1, \cdots, N-m$. $(N-m)-(m+1) \geq m-1$; i.e., $N \geq 3m$.

The standard trigonometric relation which is used in deriving the balance relation,

$$\mu_0 + \sum_{k=1}^{m-1} \mu_p \cos(pw_k) = \cos(mw_k) \tag{8.12}$$

helps in obtaining the frequencies w_k. This could be formed as mth degree algebraic equation in $\cos w$:

$$\mathcal{D}_m(\cos w)^m + \mathcal{D}_{m-1}(\cos w)^{m-1} + \cdots + \mathcal{D}_1(\cos w) + \mathcal{D}_0 = 0, \tag{8.13}$$

where $\mathcal{D}_i$, $i = 0, 1, \cdots, m$ are the functions of μ_p.

Substituting the values of μ_p, the above equation can be solved for m frequencies w_k of harmonics by using the standard numerical techniques. Various combinations of the harmonic components are formed with the frequencies w_k. The coefficients $\mathcal{A}_0, \mathcal{A}_k$, and $\mathcal{B}_k$ are estimated for each combination by using the least-squares technique. The best combination as an optimal trend is selected according to the value of the balance criterion.

The algorithm functions as below:

The discrete data is to be supplied as training set A and testing set B; one can allot a separate checking set C for examining the final optimal trend; i.e., $N = N_A + N_B + N_C$. The maximum number of harmonics is chosen as M_{max} ($< N/3$). The coefficients μ_p are estimated by using the least squares technique by forming the balance equations with the training set. The system of equations has the form:

$$\sum_{p=0}^{m-1} \mu_p [y(i+p)+y(i-p)] = y(i+m)+y(i-m);$$
$$i = m+1, \cdots, N_A - m. \tag{8.14}$$

By substituting the values of μ_p in the above mth order polynomial in $\cos w$, the frequencies are estimated; the m roots of the polynomial uniquely determine the m frequencies w_k. These frequencies are fed through the input layer of multilayer structure where the complete sifting

of harmonic trends would take place according to the inductive principle of self-organization. This is done by a successive increase in the number of terms of the harmonic components $m = 1$, $m = 2$, $m = 3, \cdots$ until $m = M_{max}$. The linear normal equations are constructed in the first layer for any $1 \leq m \leq M_{max}$ number of harmonics. The coefficients $\mathcal{A}_0, \mathcal{A}_k$, and $\mathcal{B}_k$ are estimated for all the combinations based on the training set using the least squares technique; the balance functions are then evaluated. The best trends are selected. The output error residuals of the best trends are fed forward as inputs to the second layer. This procedure is repeated in all subsequent layers. The complexity of the model increases layer by layer as long as the value of the "imbalance" decreases. The optimal trend is the total combination of the harmonical components obtained from the layers. The performance of the optimal trend is tested on the checking set C.

The program listing and sample outputs for an example are given below.

3.1 Program listing

```
C
C**********************************************************************
C THIS PROGRAM IS THE RESULT OF EFFORTS FROM VARIOUS GRADUATE STUDENTS
C  AND RESEARCH PROFESSIONALS AT THE COMBINED CONTROL SYSTEMS GROUP OF
C  INSTITUTE OF CYBERNETICS, KIEV (UKRAINE)
C**********************************************************************
C
C     HARMONICAL INDUCTIVE LEARNING ALGORITHM
C
C
C       N - NO.OF TRAINING SET POINTS
C       NP - NO.OF TEST SET POINTS
C       NE - NO.OF EXAMIN SET POINTS
C       PT - NO.OF PREDICTION POINTS
C       JFM - MAX NO.OF FREQUENCIES
C       JF - FREEDOM OF CHOICE
C       NRM - NO.OF SERIES IN HARMONICAL TREND
C       NN = N+NP+NE
C       NPT = NN+PT
C       G(NN) - DISCRETE SIGNAL DATA
C       APR(NPT) - HARMONICAL MODEL VALUES
C       MA - NO.OF LAG POINTS FOR SMOOTHING PROCEDURE(MOVING AVERAGE
C             VALUE). IF IT IS ONE, DATA REMAINS SAME
C
C**********************************************************************
C   MAIN PROGRAM
C
        INTEGER PT
        DIMENSION GY(120)
        COMMON /AB/G(120)
C
          OPEN(3,FILE='output.dat')
          OPEN(8,FILE='ts.dat')
C
        WRITE(3,4)
4       FORMAT(5X,'       L A Y E R E D  HARMONICAL ALGORITHM'/)
C
        WRITE(*,110)
110      FORMAT(3X,'GIVE NO.OF TRAIN, TEST & EXAM PTS?')
        READ(*,*)N,NP,NE
        NN=N+NP+NE
        WRITE(*,112)
```