Polynomial harmonic GMDH learning networks for time series modeling

Nikolay Y. Nikolaev, a,*, Hitoshi Iba b

a Department of Computing, Goldsmiths College, University of London, New Cross, London SE14 6NW, UK
b Department of Information and Communication Engineering, School of Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

Received 8 October 2001; accepted 20 May 2003

Abstract

This paper presents a constructive approach to neural network modeling of polynomial harmonic functions. This is an approach to growing higher-order networks like these build by the multilayer GMDH algorithm using activation polynomials. Two contributions for enhancement of the neural network learning are offered: (1) extending the expressive power of the network representation with another compositional scheme for combining polynomial terms and harmonics obtained analytically from the data; (2) space improving the higher-order network performance with a backpropagation algorithm for further gradient descent learning of the weights, initialized by least squares fitting during the growing phase. Empirical results show that the polynomial harmonic version \( ph \)GMDH outperforms the previous GMDH, a Neurofuzzy GMDH and traditional MLP neural networks on time series modeling tasks. Applying next backpropagation training helps to achieve superior polynomial network performances.

© 2003 Elsevier Ltd. All rights reserved.

Keywords: Neural network; Trigonometric function; Backpropagation training

1. Introduction

Time series from observations of natural phenomena, in the fields of ecology, meteorology, financial forecasting, etc. usually exhibit an oscillating character. The oscillations are often described by trigonometric functions which are sums of harmonics from the corresponding Fourier expansions. Such series modeling may be performed by trigonometric neural networks that employ trigonometric activation functions, usually a cosine squasher, in the hidden units. From a practical point of view, these networks are attractive due to the high accuracy of the approximations that they produce. From a theoretical point of view, these networks are reliable as they posses the universal approximation properties of the Fourier series (Cotter, 1990; Gallant & White, 1992).

Trigonometric function modeling involves determination of the harmonic frequencies, harmonic amplitudes and identification of the weights. The design of trigonometric neural networks addresses these issues in different ways: (1) training networks to identify their harmonic amplitudes assuming integer (multiple) frequencies (Gallant & White, 1992; Megson, 1993; Sanger, 1991; Zhang, Zhang, & Fulcher, 2000); (2) setting random frequencies, and estimating the weights by least squares fitting of the series pre-processed using the discrete Fourier transform (Citterio, Pelagotti, Piuri, & Rocca, 1999); (3) learning the frequencies, amplitudes, phases and weights in networks by gradient descent methods (Moody & Yarvin, 1992; Sakhnini, Manry, & Chandrasekaran, 1999); and (4) analytic derivation of unknown non-multiple frequencies and least squares estimation of the weights in growing networks (Madala & Ivakhnenko, 1994). The last two approaches are more general since real-world time series do not feature exactly periodic oscillations and need descriptions by harmonics with not necessarily proportional, non-multiple frequencies which are not known in advance. The unknown nature of the harmonics makes it necessary not only to find their non-related frequencies by analytical or connectionist learning procedures, but also to find which of the harmonics should enter the trigonometric network model since their descriptive significance is uncertain.
Constructive neural network algorithms offer search mechanisms for learning the neural network architecture. Such are the learning network algorithms from the group method of data handling (GMDH) (Ivakhnenko, 1971; Madala & Ivakhnenko, 1994). The multilayer GMDH builds higher-order polynomial networks which are interpreted as discrete Volterra models. These polynomial networks are closely related to neural network implementations of Volterra models (Marmarelis & Zhao, 1994), and to multilayer perceptrons with polynomial activation functions (Chen & Manry, 1991), PNETTR (Barron & Barron, 1988), SONN (Tenorio & Lee, 1990), VPBF (Liu, Kadirkamanathan, & Billings, 1998), ASPN (Elder & Brown, 2000). Although GMDH and the last four algorithms search for the network structure by hill climbing, they lead to models which often do not exhibit the desired performance in the sense of desired accuracy of fitting and generalization.

This paper develops a polynomial harmonic network that learns hybrids of polynomial and harmonic terms, that is models containing both time-domain and frequency-domain components. Two contributions for enhancement of the higher-order network learning are provided. First, another compositional scheme for combining polynomial terms and harmonics is offered for extending the expressive power of the network representation. According to this scheme harmonics enter activation polynomials as variables. The rationale is to make a hybrid representation for better non-linear time series modeling as the polynomials capture monotonic series curvatures well, while the harmonics capture oscillating curvatures well. The harmonic inputs are passed to a polynomial network whose architecture is built using the multilayer GMDH algorithm.

Second, an iterative gradient descent training algorithm is offered for improving the performance of polynomial neural networks. Pursuing increased accuracy in learning of the weights, a backpropagation algorithm is derived for higher-order networks with polynomial activation functions in the spirit of the feedforward neural networks theory (Rumelhart, Hinton, & Williams, 1986). The network learning process in this version, called BP-polGMDH, is decomposed into two steps: finding the function/network structure by the multilayer GMDH algorithm, and next finding accurate weights by the error backpropagation algorithm. In this sense, the suggestion is that constructive polynomial network algorithms like GMDH (Ivakhnenko, 1971), PNETTR (Barron & Barron, 1988), SONN (Tenorio & Lee, 1990), VPBF (Liu et al., 1998), and ASPN (Elder & Brown, 2000) need further weight refinement by gradient descent methods. In order to prevent network overfitting with the given series and to attain satisfactory generalization performance the search for proper polynomial as well as polynomial harmonic network structures is guided by a regularized model selection criterion using a weight decay regularization technique (Bishop, 1995; Hertz, Krogh, & Palmer, 1991).

Computer experiments with two benchmark time series are carried out. The results show that BP-GMDH is better than GMDH, phGMDH is better than GMDH but not better than BP-GMDH, and BP-phGMDH outperforms all other versions, as well as the Neurofuzzy GMDH (Ichihashi & Türksen, 1993; Ohtani, Ichihashi, Miyoshi, & Nagasaka, 1998) with Gaussian membership functions and some other neural networks, on the chosen time series.

This paper continues with the basics of polynomial harmonic approximation in Section 2. Section 3 presents the polynomial harmonic phGMDH including the identification of the unknown frequencies and the harmonic coefficients. The regularized model selection is given in Section 4. Section 5 offers the backpropagation algorithm for polynomial networks. Section 6 provides the experimental results on the chosen benchmark time series modeling tasks. Finally, a discussion is given and conclusions are derived.

### 2. Polynomial harmonic approximation

The approximation problem can be formulated as follows. Given a data series \( \mathcal{D} = \{(x_t, y_t)\}_{t=1}^N \) of vectors \( x_t \) from real data \( x \in \mathcal{M} \), and corresponding values \( y_t \in \mathcal{M} \), the goal is to find the best function model \( y = f(x) \), \( f \in L_2^1 \), which on average converges to the true unknown mapping \( \hat{f}(x) \). Time series are often described by high-order multivariate polynomials which belong to the class of discrete Volterra models (Schetzen, 1980)

\[
F(x, t) = a_0 + \sum_i a_i \varphi_i(x, t) + \sum_j \sum_k a_{ijk} \varphi_{ijk}(x, t) + \cdots
\]

(1)

where \( a_i \) are term coefficients (weights), \( x \) is an input vector \( x = (x_{-d}, x_{-d-1}, \ldots, x_{t-1}) \), \( d \) is the input dimension, and \( \varphi_i(x, t), \varphi_{ij}(x, t), \varphi_{ijk}(x, t) \ldots \) are functions of first, second, third, etc. order (degree). It is assumed that the observations \( x_{t-d}, \ldots, x_{t-1} \) are recorded at discrete time intervals \( t = 1, 2, \ldots, N \) with \( \Delta t = 1 \).

The Weierstrass theorem shows that these polynomials are a universal format for non-linear function modeling as they can approximate any continuous function on a compact set to an arbitrary precision, in an average squared residual (ASR) sense, if there is a sufficient number of terms. In practice the polynomials are truncated by design decisions so that they contain a finite number of terms.

---

1. The linear space \( L_2^1 \) contains functions with integrable squares, that is the integral: \( \int f^2(x)dx \mu \), where \( \mu \) is the space metric, exists and it is finite (Kolmogorov & Fomin, 1999).
2.1. Polynomial, harmonic and hybrid terms

There are different approaches to the selection of simple functions \( \varphi_i(x, t) \), \( \varphi_j(x, t) \), \( \varphi_{ij}(x, t) \), ... that build the polynomial models \( F(x, t) \). Most neural network approaches use polynomial terms \( P_i(x, t) = \varphi_i(x, t) \) defined as first-order univariate functions of the input variables

\[
\varphi_i(x, t) = P_i(x, t) = x_{t-i}
\]

(2)

where \( x_{t-i} \) is the variable values \( i \) time units behind \( t \), \( t \leq d \).

When real world time series featuring oscillating characteristics are provided, one should consider power series terms as well as trigonometric terms (Eubank, 1999; Graybill, 1976). The trigonometric terms, also called harmonic terms \( H_i(t) \), can be written as cosine waves in the following way

\[
\varphi_i(x, t) = H_i(t) = C_i \cos(w_i t - \phi_i)
\]

(3)

where \( i \) is the harmonic number \( 1 \leq i \leq h \), \( C_i \) is the real-value harmonic amplitude, \( w_i \) is the harmonic frequency \( 0 < w_i < \pi \), such that \( w_k \neq w_j \) for \( k \neq j \), and \( \phi_i \) is the phase angle. In case of multiple frequencies, i.e. \( w_i = 2\pi i/N \), the harmonics are \( h = (N-1)/2 \) when \( N \) is odd, and \( h = N/2 \) when \( N \) is even.

We propose harmonics to enter polynomial terms as variables through polynomial or harmonic terms \( \varphi_i(x, t) \) Eqs. (2) and (3), and through hybrid terms \( \varphi_j(x, t) \), \( \varphi_{jk}(x, t) \), ... The hybrid terms \( \varphi_j(x, t) \), \( \varphi_{jk}(x, t) \), ... in \( F(x, t) \) are high-order monomials that consist of up to \( d \) simple polynomial (2) and/or harmonic (3) functions as variables

\[
\varphi_{jk...n}(x, t) = \sum_{i=1}^{m} \varphi_i(x, t)^{r_i}
\]

(4)

where \( \varphi_i(x, t) \) is either of the simple functions defined by Eqs. (2) and (3), \( r_i = 0, 1, ... \) are the powers with which the \( i \)th element \( \varphi_i(x, t) \) participates in the \( jk...n \)th term, and the number \( m \) of functions \( \varphi \) satisfies \( 2 \leq m \leq d \).

The powers \( r_i \) are bounded by a maximum order \( r_{max} : \sum_{i=1}^{d} r_i \leq r_{max} \) for every \( jk...n \).

2.2. Hybrid function approximation

The best approximation \( F(x, t) \) of the true mapping \( \tilde{f}(x) \) minimizes the distance: \( \| \tilde{f}(x) - F(x, t) \|^2 \), where \( \| \cdot \| \) is the norm of the linear space \( L_2 \) defined: \( \| f \|^2 = (\int |f|^2 d\mu)^{1/2} \). The search for this best approximation is performed using the ASR

\[
\text{ASR} = \frac{1}{N} \sum_{i=1}^{N} (y_i - F(x_i, t))^2
\]

(5)

where \( y_i \) is the given outcome from the \( r \)th input vector \( x_i = (x_{t-1}, x_{t-j-1}, ..., x_{t-1}) \) in the series \( \mathcal{D} = \{ (x_i, y_i) \}_{i=1}^{N} \), \( F(x_i, t) \) is the outcome estimated with the same \( r \)th vector \( x_i \), and \( N \) is the series size.

3. Polynomial harmonic phGMDH

3.1. GMDH polynomial networks

The multilayer GMDH network algorithm (Ivakhnenko, 1971; Madala & Ivakhnenko, 1994) constructs hierarchical cascades of bivariate activation polynomials in the nodes, and variables in the leaves (Table 1). The activation polynomial outcomes are fed forward to their parent nodes, where partial polynomial models are made. Thus, the algorithm produces high-order multivariate polynomials by composing simple and tractable activation polynomial allocated in the hidden nodes of the network.

In neural network parlance, the higher-order polynomial networks grown by the GMDH algorithm are essentially feed-forward, multi-layered neural networks. The nodes are hidden, the leaves are inputs, and the activation polynomial coefficients are weights. The weights arriving at a particular hidden node are estimated by ordinary least squares (OLS) fitting.

There could be summarized that the GMDH-type polynomial networks influence the contemporary artificial neural network algorithms with several advantages: (1) they offer adaptive network representations that can be tailored to the given task; (2) they learn the weights rapidly in a single step by standard OLS fitting which eliminates the need to search for their values, and which guarantees finding locally good weights due to the reliability of the fitting

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Multilayer GMDH algorithm for growing higher-order networks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialization</strong></td>
<td>Data ( \mathcal{D} = {(x_i, y_i)}<em>{i=1}^{N} ), ( x_i = (x</em>{t-1}, x_{t-j-1}, ..., x_{t-1}) ), and ( k = d )</td>
</tr>
<tr>
<td><strong>Network construction and weight training</strong></td>
<td>Let the network width is ( K ), ( K &lt; c = (d-1)/2 ), the layer is ( l = 1 ), the lowest error is ( e = \text{MaxInf} ), and the activation polynomials are ( p(x_i, x_j) = a_{00} + a_{1}x_i + a_{2}x_j + a_{3}x_ix_j + a_{4}x_i^2 + a_{5}x_j^2 ) or ( p(x_i, x_j) = h_i(x_i)x_j ), so that: ( h = [h_1, h_2, ..., h_c]^T )</td>
</tr>
</tbody>
</table>

(a) Make all \( c \) combinations of variables \( (x_i, x_j) \), \( l \leq i, j \leq k \)

(b) Make a polynomial \( p_1^{c}(x_i, x_j) \) from each combination

Estimate its weights \( w_i \) by OLS fitting: \( a_1 = (H^T H)^{-1}H^T y \)

Evaluate the error of the polynomial \( p_1^{c}(x_i, x_j) = h_i \text{ ASR} = (1/N) \sum_{i=1}^{N} (y_i - p_1^{c}(x_i))^2 \)

Compute the model selection criterion \( RAE = f(\text{ASR}) \)

(c) Order the polynomials with respect to their RAE, and choose these \( k < c, 1 \leq k \leq K \), with lower criterion values

(d) Consider the lowest error from this layer: \( e^{l+1} = \min\{RAE_1\} \)

(e) If \( e^{l+1} > e \) then terminate, else set \( e = e^{l+1} \) and continue

(f) The polynomial outputs become current variables: \( x_i = p_i^{c}, 1 \leq c \leq k = K \), and \( c = K(k-1)/2 \)

(g) Repeat the construction and training step with \( l = l + 1 \)
technique; (3) these polynomial networks feature sparse connectivity which means that the best discovered networks can be trained fast by a backpropagation algorithm.

3.2. Rationale for hybrid modeling

Experimental studies revealed that the multilayer GMDH often underperforms on non-parametric regression tasks (Green, Reichelt, & Bradbury, 1988). Our research also found that on time series modeling GMDH exhibits a tendency to find very complex polynomials that cannot model well future, unseen oscillations of the series. We extend the algorithm here with harmonic terms as input variables in addition to the regular lagged variables (Fig. 1). The rationale for using polynomials and harmonics together in the target model is (1) polynomials are taken as they approximate better the monotonic curvatures as well as the discrepancies and gaps in the time series; (2) harmonics are taken as they approximate better oscillating components, spikes, and critical changes in the series curvature. Overall the polynomial harmonic models are polynomials using harmonic components, that is they are different from the traditional trigonometric polynomials (Burden & Faires, 1997).

A simple low-order bivariate activation polynomial $p(x_1, x_2)$ is considered: $p(x_1, x_2) = a_0 + a_1 x_1 + a_2 x_2 + a_3 x_1 x_2$, which leads to the following two cases: (1) polynomial and harmonic cross-product: in this case the polynomial curvatures are transferred to the harmonic, that is they amplify the spectrum of the harmonic in places determined by the polynomial; and (2) harmonic and harmonic cross-product: in this case the product curvature will contain new frequency components from the spectra of the harmonics. In both cases, the frequency spectrum of the original harmonic is modified to contain additional variances induced by the polynomial or by the harmonic with which it is multiplied.

3.3. The harmonic terms

The real world oscillating data are not exactly periodic, they exhibit slightly periodic tendencies, and in the general case require descriptions by harmonics with unknown frequencies. There are two possibilities to consider when one tries to find which harmonics should enter the model: (1) periodically oscillating series with repeating characteristics, which assume descriptions by sums of harmonically connected components $H_i(t)$ (3) with multiple frequencies: $w_i = 2\pi i/N$, $1 \leq i \leq h$; and (2) aperiodically oscillating series without repeating characteristics, that can be expressed by sums of harmonics $H_i(t)$ (3) with non-multiple frequencies $w_i$. Having non-multiple frequencies means that the sum of harmonics is not a periodic function of $t$, since the harmonics with arbitrary frequencies are not necessarily related, that is they are not necessarily commensurable. The basis functions sin and cos applied with non-multiple frequencies are not orthogonal, and the attempts to model the data in this case can be done by search for these

![Fig. 1. Polynomial harmonic GMDH network.](image-url)
harmonics that build the most close function to the true one \( f(x) \).

We develop function representations with harmonic terms with non-multiple frequencies isolated using the discrete Fourier transform. The underlying assumption is that a trigonometric function is a sum of harmonics

\[
T(t) = \sum_{i=1}^{h} [A_i \sin(w_i t) + B_i \cos(w_i t)]
\]  

(6)

where \( A_i \) and \( B_i \) are the real-value harmonic amplitudes, and the number of harmonics \( h \) is bounded by \( \max(h) \leq N/3 \), since three unknowns \( w_i, A_i, B_i \) are involved.

The possible harmonics to enter the model are derived analytically in two steps (Hildebrand, 1987, page 462): (1) calculation of the non-multiple approximate frequencies \( w_i, 1 \leq i \leq h \), of each harmonic \( i \); and (2) estimation of the coefficients \( A_i \) and \( B_i \), which enables us to determine the amplitudes \( C_i \) and the phases \( \phi_i \) for computing the separate harmonics using the concise equation \( C_i \cos(w_it - \phi_i) \).

3.4. Calculation of the non-multiple frequencies

The non-multiple frequencies \( w_i \) can be determined from an \( N \)th degree algebraic equation which is derived from \( T(t) \) (6) (Appendix A). The derivation includes finding weighting coefficients \( \alpha_q \in \mathbb{R} \) using lagged series values, substituting these coefficients \( \alpha_q \) to instantiate the algebraic equation and then solving it for \( w_i \) (Hildebrand, 1987; Madala & Ivakhnenko, 1994). The weighting coefficients \( \alpha_q, 0 \leq q \leq h - 1 \) are estimated by solving the following system of \( N - 2h \) equations by applying the least squares technique:

\[
\sum_{q=0}^{h-1} \alpha_q(y_{t+q} + y_{t-q}) = y_{t+h} + y_{t-h}
\]

(7)

where \( y_t \) denotes the \( t \)th value from the given series, and the range is \( t = h + 1, \ldots, N - h \). These coefficients \( \alpha_q \) are used to instantiate the equation for the frequencies \( w_i \) as follows:

\[
\alpha_0 + \sum_{q=1}^{h-1} \alpha_q \cos(w_i q) = \cos(w_i h),
\]

(8)

for \( w_i, 1 \leq i \leq h \)

After expressing all \( \cos(w) \) as polynomials of degree \( i \) in \( \cos(w) \), Eq. (8) becomes an \( h \)th degree algebraic equation in \( \cos(w) \) for the non-multiple frequencies (Hildebrand, 1987)

\[
\alpha_0' + \alpha_1' \cos(w) + \alpha_2' \cos^2(w) + \cdots + \alpha_h' \cos^h(w) = 0
\]

(9)

where the new coefficients \( \alpha_q' \) result from Eq. (8) as functions of the coefficients \( \alpha_q, 0 \leq q \leq h - 1 \).

Eq. (9) is of the kind \( g(w) = 0 \) and can be solved by the Newton–Raphson method (Burden & Faires, 1997). Thus, \( h \) approximate roots are found which are the frequencies \( w_i, 1 \leq i \leq h \), of the \( h \) harmonics. Among these calculated \( h \) roots for \( \cos(w) \) the admissible values are those that lie between \(-1 \) and \( 1 \), since \( |\cos(w)| \leq 1 \), from frequencies \( 0 < w < \pi \). Solving Eq. (9) is of critical importance for the success of polynomial harmonic neural network modeling since it influences the richness of the available frequency spectrum, and determines the harmonics that will be available for further learning.

3.5. Isolation of significant harmonics

Among all harmonics with non-multiple frequencies only those harmonics which are most statistically significant for describing the time series should be considered. The significant harmonics can be identified by drawing periodograms with plots of the intensity function (Kendall & Ord, 1983)

\[
I(w_i) = \frac{N(A_i^2 + B_i^2)}{4\pi}
\]

(10)

where \( A_i \) and \( B_i \) are the coefficients of the \( i \)th harmonic with frequency \( w_i \).

3.6. Computing of the harmonics

In case of non-multiple frequencies the trigonometric models \( T(t) \) (6) are linear in the coefficients sums of harmonics, that is these are linear models of the kind \( Tc = y \). The amplitudes \( c = (b_0, A_1, B_1, A_2, B_2, \ldots, A_h, B_h) \) are found by solving the normal trigonometric equation: \( c = (T^T T)^{-1} T^T Y \) (Appendix A). After that, the amplitudes \( C_i \) and phases \( \phi_i \) are computed from the formulae

\[
C_i = \sqrt{A_i^2 + B_i^2}, \quad \phi_i = \arctan(B_i/A_i)
\]

(11)

where \( i \) denotes the concrete harmonic number.

4. Regularized model selection

A model selection criterion is necessary to achieve overfitting avoidance, that is to pursue construction of not only accurate but also predictive networks. The model selection criterion is essential since it guides the construction of the network topology, and so influences the quality of the induced function model. Two primary issues in the design of a model selection function for overfitting avoidance are: (1) favoring more fit networks by incorporating a mean-squared-error sub-criterion; and (2) tolerating smoother network mappings having higher generalization potential by incorporating a regularization sub-criterion.

Knowing that a large weight in a term significantly affects the polynomial surface curvature in the dimensions determined by the term variables, a correcting smoothness sub-criterion that accounts for the weights’ magnitude is accommodated in a regularized average error RAE as
where $a$ is the weights, $1 \leq j \leq W$. This formula (12) is known as weight decay regularization (Bishop, 1995; Hertz et al., 1991). This weight decay regularization requires to use the regularized least squares (RLS) fitting method for estimating the weights:

$$
a = (X^T X + \lambda I)^{-1} X^T y$$

where $a$ is the $(m + 1) \times 1$ coefficients vector, $X$ is $N \times (m + 1)$ matrix of row vectors $(1, x_1, x_2, \ldots, x_N)$, $N \times 1$ output vector, and $\lambda$ is the regularization parameter. Proper values for the regularization parameter $\lambda$ are found using statistical techniques (Myers, 1994).

5. Backpropagation for polynomial networks

The multilayer GMDH estimates the weights of the activation polynomials while growing the network architecture. This, however, does not guarantee reaching the desired characteristics mainly because the weights are not sufficiently coordinated within the final polynomial network. As a remedy to this problem, we propose to perform additional weight improvement by an error backpropagation algorithm for higher-order networks with polynomial activation functions (Rumelhart et al., 1986).

5.1. Motivation

The key idea is to improve further the weights found by constructive polynomial network algorithms. Starting with the network architecture and weights computed during network growing, not with random weights, the error of fit is backpropagated from the output layer to the input layer in order to tune low layer\(^2\) weights so that they capture the data series characteristics in coherence with the higher layer network weights. This is necessary because the growing network algorithms usually estimate the weights from input to output layers in such a way that the computed lower layer weights do not depend on the higher layer weights. That is why, the lower layer weights are not necessarily in coordination with the higher layer weights.

Polynomials can be used directly as activation functions in neural networks since they are both (1) non-linear, which gives possibility for representing highly non-linear functions; and (2) continuous, so they can be differentiated to enable gradient descent learning. The polynomials are more suitable than the sigmoid as activation functions for the hidden network nodes because they describe non-linear mappings more flexibly and lead to more parsimonious topologies (Marmarelis, 1994).

5.2. The delta learning rules

The backpropagation training algorithm for polynomial networks iteratively decreases the output network error by adjusting the weights according to the gradient descent technique. This technique is implemented by the delta learning rule given by the formula: $a = a + \Delta a$, where $\Delta a$ is the weight update $\Delta a = -\eta \Delta E_f / \partial a$, $\eta$ is the learning rate, and $E_f = (1/2)(y_i - F_i)^2$ is the computed error between the given output $y_i$ and the estimated neural network output $F_i$.

$$(1) \ 1 \leq t \leq N.$$ Since in GMDH networks the activation polynomial $p(x_1, x_2) = a_0 + a_1 x_1 + a_2 x_2 + a_3 x_1 x_2$ actually plays the role of both activation function $p = f(p)$ and net-input function the error derivatives are: $\partial E_f / \partial a = (\partial E_f / \partial p)(\partial p / \partial a)$. The second part $\partial p / \partial a = f'$ is the error change as a function of a weight $a$ at that node. In case of GMDH networks there are three partial derivatives:

$$x'_1 = \frac{\partial p}{\partial a_1} = x_1, \quad x'_2 = \frac{\partial p}{\partial a_2} = x_2, \quad x'_3 = \frac{\partial p}{\partial a_3} = x_1 x_2$$

There are two derivatives of the kind $\partial E_f / \partial p$ : one for the output nodes and one for hidden nodes.

Delta rule for output node weights. Suppose that the output nodes are indexed by $k$ and the hidden nodes that feed them are indexed by $j$. Then, the delta rule for the output nodes becomes:

$$\Delta a_{jk} = -\eta \partial E_f / \partial a_{jk} = \eta \beta_k x'_j$$

where $\beta_k = (y_i - F_i)$. In BP-phGMDH $\Delta a_{jk}$ should be instantiated four times for the four weights: $a_0, a_1, a_2, a_3$.

Delta rule for hidden node weights. Let us assume that the hidden nodes are indexed by $j$ as above and the hidden nodes that feed them are indexed by $i$. Then, the delta rule for the hidden nodes becomes

$$\Delta a_{ij} = -\eta \partial E_f / \partial a_{ij} = \eta \beta_p x'_i$$

where $\beta_p = \sum_j (-\beta_k)p_j$, and $p_j$ is the polynomial outcome from node $i$. In BP-phGMDH $p_i = a_1 + a_3 x_2$ in case of feeding node $j$ with input $x_1$, or, respectively, $p_i = a_2 + a_3 x_1$ in case of feeding node $j$ with input $x_2$. Note that the inputs $x_1$ and $x_2$ to a node can be outputs from the lower layer nodes: $x_1 = p$, $x_2 = p$ or input variables directly: $x_i$ or $C_i \cos(\omega t - \phi_j)$.

6. Time series modeling by polynomial networks

The developed polynomial harmonic network is tested on the time series prediction problems. Input vectors $\mathbf{x}$ are created with embedding dimension $d$, and delay time $\tau$:
Table 2
Computed parameters: frequencies, amplitudes and phases, for the four harmonics identified from the Mackey–Glass equation series, used in phGMDH and BP-phGMDH to produce the results given in Table 3

<table>
<thead>
<tr>
<th>Harmonic</th>
<th>$w_i$</th>
<th>$A_i$</th>
<th>$B_i$</th>
<th>$C_i$</th>
<th>$\phi_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.141954</td>
<td>-0.153170</td>
<td>0.033915</td>
<td>0.156880</td>
<td>-0.217906</td>
</tr>
<tr>
<td>2</td>
<td>0.173460</td>
<td>-0.023613</td>
<td>0.058528</td>
<td>0.063112</td>
<td>-1.187317</td>
</tr>
<tr>
<td>3</td>
<td>0.242038</td>
<td>-0.015695</td>
<td>0.027781</td>
<td>0.031908</td>
<td>-1.056531</td>
</tr>
<tr>
<td>4</td>
<td>0.372572</td>
<td>-0.003557</td>
<td>0.013278</td>
<td>0.013746</td>
<td>-1.309055</td>
</tr>
</tbody>
</table>

$x_t = (x_{t-5}, x_{t-(d-1)5}, \ldots, x_{t-1})$. The dependent variable is the immediate next point $x_t$. The series values $x_t$ are normalized to have zero mean and unit variance.

Five network versions are investigated: the multilayer GMDH (Ivakhnenko, 1971), its version improved by backpropagation BP-GMDH, the polynomial harmonic phGMDH, its version improved by backpropagation BP-phGMDH, and a Neurofuzzy GMDH with Gaussian membership functions (Ichihashi & Türkmen, 1993; Ohtani et al., 1998). These bottom-up constructive algorithms from the GMDH family are related to the top-down constructive network algorithm LMS Tree (Sanger, 1991) made to produce trigonometric polynomials.

The Neurofuzzy GMDH in all experiments have been designed with three layers having four nodes each. Every node performs weighted summation of four multiplied pairs of Gaussian membership functions: $\exp(-((x_{t-i} - c)^2)/s)$ applied to distinct inputs. The Gaussian functions are initialized with equal spread parameters $s = 0.5$, and uniformly placed center parameters as follows (Ohtani et al., 1998): $c_{00} = 0.0$, $c_{01} = 0.0$, $c_{10} = 0.0$, $c_{11} = 1.0$, $c_{20} = 1.0$, $c_{21} = 0.0$, $c_{30} = 1.0$, and $c_{31} = 1.0$. The weights as well as the local center and spread parameters of the Neurofuzzy GMDH have been trained with a localized fuzzy backpropagation algorithm using a learning rate $\eta_t = 0.02$ (without using momentum) for up to 300 epochs.

The LMS Tree algorithm builds function series expansions of one-dimensional basis functions $\varphi(x, t)$. The implementation of Sanger (1991) is considered using single variable polynomials $\varphi(x, t) = x_{t-i}$ and trigonometric basis functions $\varphi(x_{t-i}, t) = \sin(2\pi t x_{t-i}/r)$, $\varphi(x_{t-i}, t) = \cos(2\pi t x_{t-i}/r)$, where $r = b - a$, $x \in [a, b]$. LMS Tree grows function expansions by generating the tree model in top-down manner expanding the child node linked to its parent by the connection with largest weight variance. The learning involves constructing of the network tree, and identification of the weights using the Least Mean Squares rule.

The accuracy and generalization potential of the derived solutions are measured using the average relative variance (ARV) formula (Weigend, Huberman, & Rumelhart, 1992). The ARV values are computed always from the beginning of the series in order to become consistent with the reported results from other research.

6.1. Processing the Mackey–Glass series

A trajectory of 400 points is produced with the Mackey–Glass (Mackey & Glass, 1977) differential equation using parameters: $a = 0.2$, $b = 0.1$, and differential $\Delta = 17$. Embedding dimension $d = 10$ and delay time $\tau = 1$ are selected heuristically. The first 100 points are used for training, and the remaining for testing.

Initially, 10 harmonics were identified analytically from the training series using the approach to calculating the frequencies given in Section 3.4. Next, plots of the intensity function, produced using formula (10), against the frequency were made and only four significant harmonics were taken. The computed frequencies $w_i$, amplitudes $A_i$, $B_i$, $C_i$, and phases $\phi_i$, $1 \leq i \leq 4$, of the selected four harmonics are given in Table 2. The remaining inputs from the embedding dimension are filled with six independent variables. Thus, the 10 variables passed to the model form input vectors of the kind: $\mathbf{x} = (x_{t-1}, x_{t-2}, \ldots, x_{t-5}, x_{t-6}, H_1, H_2, \ldots, H_4)$. Fig. 2
displays these variables and harmonics. An approximated segment from the Mackey–Glass curve by the trained and improved BP-\(ph\)GMDH system is plotted in Fig. 3.

Several observations can be made from the results presented in Table 3. (1) The backpropagation version BP-GMDH is better at fitting and prediction than the multilayer GMDH; (2) the polynomial harmonic version \(ph\)GMDH exhibits better accuracy and higher generalization than GMDH but not to BP-GMDH; (3) the backpropagation harmonic version BP-\(ph\)GMDH is the best among all the studied algorithms for accuracy with ARV \(0–400 = 0.008254\); for short term prediction with ARV \(0–200 = 0.007181\), and for long-term prediction with ARV \(0–400 = 0.006583\); and (4) the Neurofuzzy GMDH is only slightly worse than BP-\(ph\)GMDH and better than \(ph\)GMDH.

One notes that the improvement in forecasting by BP-\(ph\)GMDH related to \(ph\)GMDH is not as high as that achieved by \(ph\)GMDH compared to GMDH, and this requires further research. The observation that the eurofuzzy GMDH exhibits very good performance is an indication that the fuzzy model tends to capture well the characteristics in the data by adapting its local parameters. From another point of view, the slight improvement in generalization on this task achieved with the novel algorithms can be attributed to the relatively smooth curvature of the considered as benchmark Mackey–Glass series.

Table 3 compares the studied versions of GMDH with the relevant learning network systems: SONN \(\text{(Tenorio \\& Lee, 1990)}\) and LMS Tree \(\text{(Sanger, 1991)}\), that also construct the network architecture and serve for function approximation. SONN \(\text{(Tenorio \\& Lee, 1990)}\) produces network representations of high-order multivariate polynomials. It builds GMDH-like networks using a simulated annealing algorithm for searching the most promising pairs of variables for the hidden units at each next layer. The best reported result \(\text{(Tenorio \\& Lee, 1990, page 106, Fig. 4)}\) was taken ready and only re-estimated: it showed relatively good predictability especially in the long-term ARV \(0–400 = 0.008950\).

The LMS Tree is tuned to learn a tree-structured network with resolution Resolution = 14: of these six are polynomial basis functions, i.e. the variables \(x_{t-1}, x_{t-2}, \ldots, x_{t-5}, x_{t-6}\), and eight are basis functions, half of them sin and half cos. The tree depth is restricted to \(q = 15\), and the learning rate is \(\eta = 0.005\). The intention is to make the inputs close to these passed to GMDH for fair comparisons. The results from LMS Tree given in Table 3 are worse than those obtained by all other analyzed network algorithms. A possible explanation is that using hybrid networks with polynomial and harmonic basis functions is not the only necessary condition for achieving satisfactory performance.

![Fig. 3. Approximated segment from the Mackey–Glass curve by a learned network with BP-\(ph\)GMDH.](image)

<table>
<thead>
<tr>
<th>Accuracy (ARV), 0–100</th>
<th>Generalization (ARV)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0–200</td>
</tr>
<tr>
<td>GMDH</td>
<td>0.009171</td>
</tr>
<tr>
<td>BP-GMDH</td>
<td>0.008368</td>
</tr>
<tr>
<td>(ph)GMDH</td>
<td>0.009029</td>
</tr>
<tr>
<td>BP-(ph)GMDH</td>
<td>0.008254</td>
</tr>
<tr>
<td>NeurofuzzyGMDH</td>
<td>0.008266</td>
</tr>
<tr>
<td>SONN</td>
<td>0.011610</td>
</tr>
<tr>
<td>LMS Tree</td>
<td>0.057961</td>
</tr>
</tbody>
</table>

Backprop training of BP-GMDH and BP-\(ph\)GMDH using learning rate \(\eta = 0.001\) during Epochs = 100. NeurofuzzyGMDH with NumberOfLayers = 3 and NodesInLayer = 4 using \(\eta' = 0.02\), and LMS Tree using NumberOfSubtrees = 5, Resolution = 14.
function approximation. The performance of constructive network algorithms with hybrid representations depends also on the implemented search method. The presented results in Table 3 and these in the following Table 5 are recorded with the heuristically selected as optimal parameters for the greedy, hill climbing search methods with which they are developed.

6.2. Processing the sunspots series

The not-exactly periodic Sunspots series (Weigend et al., 1992) contains 280 data points, divided into three subsets: these from 1700 to 1920 for training, and these from 1921 to 1955 as well as from 1956 to 1979 for testing. Initially, 10 harmonics were analytically derived from the training series using the approach from Section 3.4, and the most significant four of them \( h = 4 \) were isolated using the periodogram of the intensity function (10). The remaining slots from the embedding dimension were six input variables \( d = 6 \), sampled with a time lag \( \tau = 1 \) from the Sunspots series. Fig. 4 displays these harmonics and variables passed for learning. The numeric values of the frequencies, amplitudes and phases of these harmonics are given in Table 4.

![Fig. 4. Training variables and harmonics identified from the Sunspots series.](image_url)

The results given in Table 5 allow us to think that using the error backpropagation algorithm to adjust the weights of GMDH polynomial networks really helps to achieve improvement. The network models learned by the polynomial harmonic \( ph \)GMDH show higher accuracy and generalization potential than GMDH but underperform BP-GMDH and BP-\( ph \)GMDH. The BP-\( ph \)GMDH system is most successful again than all the remaining GMDH algorithms, demonstrating best fitting with \( ARV_{1700–1920} = 0.095818 \), lowest short-term forecast error \( ARV_{1700–1955} = 0.110784 \) in the future period 1700–1955, as well as lowest long-term forecast error \( ARV_{1700–1979} = 0.121957 \) in 1700–1979. The Neurofuzzy GMDH is again only slightly worse than BP-\( ph \)GMDH which allows us to think that it has powerful learning potential. This observation also shows that the backpropagation training in Neurofuzzy GMDH and

<table>
<thead>
<tr>
<th>Harmonic</th>
<th>( w_i )</th>
<th>( A_i )</th>
<th>( B_i )</th>
<th>( C_i )</th>
<th>( \phi_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.56792</td>
<td>0.130746</td>
<td>0.058067</td>
<td>0.143058</td>
<td>0.417966</td>
</tr>
<tr>
<td>2</td>
<td>0.55587</td>
<td>0.020158</td>
<td>0.111283</td>
<td>0.113094</td>
<td>1.391592</td>
</tr>
<tr>
<td>3</td>
<td>0.60173</td>
<td>0.01318</td>
<td>0.077300</td>
<td>0.078417</td>
<td>1.401833</td>
</tr>
<tr>
<td>4</td>
<td>0.61419</td>
<td>0.004923</td>
<td>0.043696</td>
<td>0.043973</td>
<td>1.458591</td>
</tr>
</tbody>
</table>

Table 4
Computed parameters: frequencies, amplitudes and phases, for the four harmonics identified from the Sunspots data series, used in \( ph \)GMDH and BP-\( ph \)GMDH to produce the results given in Table 5.
An approximated segment of the Sunspots series by the polynomial harmonic network further refined by backpropagation BP-phGMDH is plotted in Fig. 5. The numerical results in Table 5 confirm the expectations that using harmonic components can help to model better series with spikes, like these in Fig. 5, because the phGMDH system is better than GMDH, and BP-phGMDH is better than BP-GMDH. That is why we are inclined to think that when the series contains regular appearances of spikes, not sporadic spikes, the hybrid representations learned by phGMDH and BP-phGMDH may have considerably better approximation qualities than those of the original GMDH and BP-GMDH.

Table 5 also provides results obtained by the LMS Tree algorithm (Sanger, 1991). The LMS Tree is tuned with inputs: six basis functions which are the variables $x_{t-1}, x_{t-2}, \ldots, x_{t-5}, x_{t-6}$, and eight basis functions of which half are sin and half are cos (Resolution = 14). The tree depth is restricted to $q = 15$, and the learning rate is $\eta = 0.005$. One notes that the grown LMS Tree network exhibits worse accuracy and worst generalization than the GMDH algorithms. LMS Tree seems also worse than the standard backpropagation algorithm applied with a neural network with a fixed topology for this task. The LMS Tree shows highest fitting error $\text{ARV}_{1700–1920} = 0.215247$ in the period 1700–1920, highest short-term prediction error $\text{ARV}_{1700–1955} = 0.228952$ in 1700–1955, and long-term prediction error $\text{ARV}_{1700–1979} = 0.442850$ in 1700–1979.

The results from the investigated different polynomial and Neurofuzzy GMDH systems as well as LMS Tree are compared in Table 5 with the often cited results obtained by a fixed MLP neural network trained with the backpropagation algorithm (Weigend et al., 1992). It is interesting to observe in Table 5 that the trained MLP is better than the GMDH versions on interpolation with $\text{ARV}_{1700–1920} = 0.082$ and short-term prediction with $\text{ARV}_{1700–1955} = 0.086$ in 1700–1955, but the MLP is more than twice as bad in the ARV sense than all GMDH network models on long-term prediction. The MLP demonstrates superior performance than LMS Tree on interpolation, on short- and long-term forecasting.

7. Discussion

Theoretical advantages. The benefit of harmonic components in compositions of polynomial basis functions is in the induction of additional non-linearities in the target model. The employment of analytically discovered non-multiple frequencies extends the expressive power of the hybrid representations because when used in the cross-product terms they modify the curvatures that they imply. When the simple bivariate basis polynomial is taken, the resulting product curvatures contain new frequency components that arise as sums and differences from the participating frequencies. In addition to this, when other bases are selected such as the complete bivariate basis polynomial with quadratic variables, for example, there will appear additional spectra with multiple frequencies due to the squared sin and cos functions. Application of an algorithm like the multilayer GMDH learning network with such hybrid polynomial harmonic representations is necessary to perform learning of the proper model structure for the task.

The non-mutually related frequencies are a factor which allows us to achieve very close adaptations to aperiodically oscillating series curvatures. The expressive power of the proposed hybrid representation scheme strongly depends on the method for estimating these non-multiple frequencies.
The method adopted in this paper produces very accurate frequencies since it uses the given series data directly (Hildebrand, 1987). There is an alternative method for estimating the non-multiple frequencies relying entirely on least squares techniques (Bloomfield, 2000). According to this, the harmonic coefficients are estimated by least squares fitting the given series assuming fixed but dummy frequencies, and then the derived harmonic coefficients are considered in another least squares procedure to identify the frequencies. The method of Hildebrand (1987) is preferred as it produces more accurate frequencies than that of Bloomfield (2000). This is because Bloomfield estimates the frequencies using substitutions with the derived harmonic coefficients, and so the errors accumulated in the computation of the harmonic coefficients are augmented in the estimates of the frequencies.

**Applicability of phGMDH.** When polynomial and harmonic components are employed together for function approximation by GMDH learning networks the expected improvement of phGMDH obtained in comparison with GMDH depends to a great degree on the oscillations of the time series. Based on the empirical results given in this paper, three cases may be identified and analyzed as follows:

1. when the series contains oscillations which are not necessarily related the phGMDH may be expected to achieve small improvement over GMDH, as we found during experiments with the Mackey–Glass series;
2. when the series contains oscillations with repeating sharp spikes in their curvature the use of harmonics in the function representation may help to attain much better results by polynomial harmonic phGMDH over GMDH, as we showed with the processing of Sunspots series.

**Necessity of backpropagation learning.** The idea for applying a backpropagation weight learning algorithm for further adjustment of the polynomial networks is fundamental and proves the advantages of such gradient descent algorithms even if the network topology has been properly selected in advance. The fact that there exist constructive polynomial network algorithms does not mean that they are able to identify both the most appropriate network/model structure and the network weights. What we demonstrate using the backpropagation algorithm for networks with polynomial activation functions is that the systems from the GMDH family produce suboptimal solutions which assume a considerable further improvement.

Concerning the attained results we would like to emphasize that they have been produced using a very small learning rate. This is because the error correction by backpropagation begins with already trained weights, those estimated by least squares fitting using the network construction algorithm. These weights in the discovered polynomials are on the slopes of the error surfaces toward some optima that has not been reached, however. In order to continue the gradient descent on the slopes down toward the basin minima, very small learning rates must be used so as to avoid overshooting the minima. Note that the convergence will not be slow since the weights are to some extent trained, that is they are relatively close to the minima.

**8. Conclusion**

This paper contributes to the research into increasing the expressive power and learning efficacy of polynomial network algorithms for time series modeling. Initial results have been reported from the development of a hybrid version of the multilayer GMDH network algorithm. This hybrid phGMDH considers polynomial activation functions in the hidden nodes and polynomial as well as harmonic terms passed as input variables. The harmonic terms are computed by spectral analysis of the given series data assuming that their frequencies are non-multiple. Then search by bottom-up network growing is performed to find which harmonics should enter the model. Another contribution is the derivation of an error backpropagation weight-learning algorithm for polynomial networks which is successfully applied to networks generated by the multilayer GMDH. In order to obtain best results with such polynomial network growing algorithms, the recommendation is always to perform error backpropagation on the learned networks.

The reported experimental results allow us to hypothesize that the phGMDH and BP-phGMDH algorithms could be of practical importance. They can be used successfully for non-parametric approximation because of the following advantages: (1) they generate explicit analytical models in the form of multivariate high-order polynomial and polynomial harmonic functions amenable to human comprehension; (2) they learn the network topology, thus tailoring the network to the task and avoiding overfitting problems arising from the network architecture which is unknown in advance; and (3) they make the polynomials well-conditioned, thus computationally stable and suitable for practical purposes.

**Acknowledgements**

This research has been conducted in The University of Tokyo, where the authors were supported by Grand-in-Aid (12208004) for Scientific Research on Special Areas, Genome Informatics, from The Ministry of Education, Science, Sports and Culture of Japan.
The authors are grateful to Professor A.G. Ivakhnenko for the invaluable discussions on GMDH and on the algorithm for identification of non-multiple harmonics from a time series. We also thank Professor Ichihashi and Takashi Ohtani for the useful discussions on the Neurofuzzy GMDH and providing us the program code. Thanks also to Terrence Sanger for the implementation of the LMS Tree algorithm.

Appendix A. Derivation of the trigonometric equations

A.1. System of equations for the weighting coefficients

Eq. (7) is derived using the trigonometric function \( T(t) \) (6). The function values \( y_{t+q} \) from the given time series at arbitrary points \( t+q \), equally spaced from a fixed point \( t \), can be expressed as follows

\[
y_{t+q} = \sum_{i=1}^{h} [A_i \sin(w_i t + w_i q) + B_i \cos(w_i t + w_i q)]
\]

The rationale for using Eq. (8) in Eq. (A4) relies on the choice for simplifying Eq. (A4) using Eq. (A3) is exactly like Eq. (8) of... Producing Eq. (A4). Using formula (A5) to reduce all frequencies \( \cos(hw) \), as it leads to \( \alpha_0 + \alpha_1 \cos(w_0) + \cdots + \alpha_{h-1} \cos((h-1)w) = \cos(hw) \).

A.2. Algebraic equation for the frequencies

The reduction of all multiplied frequencies in Eq. (8) from \( \cos(iw) \), \( 1 \leq i \leq h \) to \( \cos(w) \), is done successively according to the recursive formula

\[
\cos(iw) = 2 \cos((i-1)w) \cos(w) - \cos((i-2)w)
\]

which is produced as follows:

\[
\cos(iw) = \cos((i-1)w) + w
\]

and \( \cos((i-2)w) = \cos((i-1)w) - w \)

After summation of \( \cos(iw) \) and \( \cos((i-2)w) \) we obtain \( \cos(iw) + \cos(i-2)w = 2 \cos((i-1)w) \cos(w) \)

which is easily transformed into Eq. (A5). Using formula (A5) to reduce all frequencies \( \cos(iw) \) in Eq. (8) to \( \cos(w) \), one arrives at an hth degree algebraic equation for the non-multiple frequencies presented by Eq. (9): \( \alpha_0 + \alpha_1 \cos(w) + \alpha_2 \cos^2(w) + \cdots + \alpha_h \cos^h(w) = 0 \).

A.3. The normal trigonometric equation

In order to find the harmonic amplitudes \( A_i \) and \( B_i, \) \( 1 \leq i \leq h \), assuming that they form a vector: \( \mathbf{c} = (b_0, A_1, B_1, A_2, B_2, \ldots, A_h, B_h) \), one has to solve the normal trigonometric equation \( \mathbf{c} = (\mathbf{T}^T)^{-1}\mathbf{T}^T\mathbf{Y} \) using the matrix.
which size is \(N \times (2h + 1)\) as there are \(2h\) coefficients \(A_i\) and \(B_i, 1 \leq i \leq h\). The multiplication \(T^T T\) leads to the following \((2h + 1) \times (2h + 1)\) covariance matrix

\[
\begin{bmatrix}
1 & \sin(w_1 x_1) & \cos(w_1 x_1) & \sin(w_2 x_1) & \cos(w_2 x_1) & \cdots & \sin(w_N x_1) & \cos(w_N x_1) \\
1 & \sin(w_1 x_2) & \cos(w_1 x_2) & \sin(w_2 x_2) & \cos(w_2 x_2) & \cdots & \sin(w_N x_2) & \cos(w_N x_2) \\
\vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\
1 & \sin(w_1 x_N) & \cos(w_1 x_N) & \sin(w_2 x_N) & \cos(w_2 x_N) & \cdots & \sin(w_N x_N) & \cos(w_N x_N)
\end{bmatrix}
\]  

(A8)

where the outcome vector \(Y\) contains \(N\) values \(y_t, 1 \leq t \leq N\).

\[
\begin{bmatrix}
N \sum_{i=1}^{N} \sin(w_1 x_i) & N \sum_{i=1}^{N} \cos(w_1 x_i) & \cdots & N \sum_{i=1}^{N} \cos(w_N x_i) \\
N \sum_{i=1}^{N} \sin(w_1 x_i) & N \sum_{i=1}^{N} \sin^2(w_1 x_i) & \cdots & N \sum_{i=1}^{N} \sin(w_1 x_i) \cos(w_N x_i) \\
N \sum_{i=1}^{N} \cos(w_1 x_i) & N \sum_{i=1}^{N} \cos(w_1 x_i) \sin(w_1 x_i) & \cdots & N \sum_{i=1}^{N} \cos^2(w_1 x_i) \\
\vdots & \vdots & \vdots & \vdots \\
N \sum_{i=1}^{N} \sin(w_N x_i) & N \sum_{i=1}^{N} \sin(w_1 x_i) \sin(w_N x_i) & \cdots & N \sum_{i=1}^{N} \sin(w_N x_i) \cos(w_N x_i) \\
N \sum_{i=1}^{N} \cos(w_N x_i) & N \sum_{i=1}^{N} \cos(w_N x_i) \sin(w_N x_i) & \cdots & N \sum_{i=1}^{N} \cos^2(w_N x_i)
\end{bmatrix}
\]  

(A9)

where the summations are over all \(N\) points. The vector \(T^T Y\) of size \((2h + 1) \times 1\) is:

\[
T^T Y = \begin{bmatrix}
\sum_{i=1}^{N} y_i \\
\sum_{i=1}^{N} y_i \sin(w_1 x_i) \\
\sum_{i=1}^{N} y_i \cos(w_1 x_i) \\
\vdots \\
\sum_{i=1}^{N} y_i \sin(w_N x_i) \\
\sum_{i=1}^{N} y_i \cos(w_N x_i)
\end{bmatrix}
\]  

(A10)

References


