C 23773
24.09.97

997 39

On Some Methods of Forecasting with Wavelets.

A dissertation submitted in partial fulfillment of the M.Tech.(Computer Science) degree of The Indian Statistical Institute.

by

Anirvan Basu

under the supervision of

Prof M.K Kundu and Prof N.R Pal.
Machine Intelligence Unit.
Indian Statistical Institute.

July 30, 1997

Indian Statistical Institute.
203, B.T Road.
Calcutta - 700 035.

Certificate of Approval.

Certified that the dissertation titled **"On Some Methods of Forecasting with Wavelets"** submitted by Sri **Anirvan Basu**, as partial fulfillment of the requirements for the degree of Master of Technology in Computer Science is a record of the work of the student, which has been carried out under our supervision.

[illegible] [illegible]

(Prof. M.K Kundu) **(Prof. N.R Pal)**

Machine Intelligence Unit.

Acknowledgements

I express my deepest sense of indebtedness and gratitude to my guides Prof. Malay K Kundu and Prof. Nikhil R Pal for their timely guidance and regular supervision during the entire course of my M.Tech dissertation.

I am also grateful to Prof B.B Chaudhuri and Prof. P Sengupta of the C.V.P.R Unit for extending the computational facilities of their unit for my purpose.

Among students I would like to thank Arindam Das, my classmate and Sitabhra Sinha, research scholar at M.I.U for helpful and enlightening discussions pertaining to the subject of my dissertation. I would also like to thank P.S Umesh Adiga, research scholar at the C.V.P.R Unit for helping me with the computational facilities at that unit. He took his time off to provide me with computation time at the unit's facilities.

Finally I would like to thank all others who have helped me now and then, in making my dissertation a success.

Anirvan Basu.
Dated : 28th July 1997.

Abstract

There exist various methods for function forecasting and function approximation in the Statistical disciplines. Nowadays various powerful mathematical techniques and also some adaptive techniques have been established in this domain.In the present work a conventional model and several recent models are investigated. Out of the adaptive models , the *Wavelet Network* model combines the wavelet transform method with a Backpropagation-type learning algorithm for its purpose.This model is then modified to incorporate *multiple mother wavelets* and shown to yield better results.Finally an altogether new evolutionary technique using *Genetic Algorithms* and *wavelet decomposition* is proposed and implemented. This model is then further improved using a penalty-based GA model.

1 Introduction

Time series is a sequence of observations taken sequentially in time (or some other independent parameter). Applications of time series can be found in diverse fields such as economics , business , engineering , the natural sciences(especially geophysics and meteorology) , and the social sciences. An intrinsic feature of a time series is that,typically adjacent observations are *dependent*.This dependence among observations is utilised to develop various models for forecasting and function approximation.

Generally in time series prediction there are two classes of models. One class of models are mainly concerned with long-term prediction. These models are used to predict various types of *trends* such as annual trends , seasonal trends , cyclical trends , etc.As a consequence they use various $trend - prediction$ mechanisms for their purpose. They are comparatively simpler in nature.

The other class of models are concerned with short-term prediction , such as the daily fluctuation of stock prices , the hourly fluctuation of chemical concentration in a mixture , etc. The time series themselves being very complex the models of this class are also complex in nature. They range from the relatively simple Moving Average (MA) model through the Autoregressive (AR) model to the compound ARMA and ARIMA models.A survey of most of these important models can be found in [Tere'90].

Other than these conventional models there are several unconventional ones based on recent techniques such as Artificial Neural Networks , Fuzzy Logic and Genetic Algorithms. Nowadays much emphasis is being given on such techniques. Models based on such techniques have been proven to be very powerful in function approximation and forecasting of extremely complex time series.Their power comes from the inherent non-linear nature of Neural Networks and the evolutionary nature of Genetic Algorithms.

In this work several models from the latter class are investigated. The first model discussed is the well-known *Autoregression* model[BoxJ'70]. It is a well established Statistical model and is best-suited for processes whose characteristics

(i.e., the parameters involved and their number) do not change throughout the series of observations (that have been made) and the forecasts (that will be made). However it can as well be applied to processes which do not exhibit these qualities. Among the recent non-conventional models discussed , is a recent method based on Wavelet Transforms [Zhan'92]. This model is investigated in detail and its drawbacks and limitations are then highlighted.Then a modified version of this model is proposed and experimented. Finally an altogether new evolutionary technique is developed which combines the power of *Genetic Algorithms* with wavelet decomposition for function forecasting.

The organisation of the article is as follows :
In section 2 the Autoregressive model is discussed. In this section two separate methods for estimating the parameters of this model are discussed. Finally the Akaike Information criterion is discussed in the context of model selection criteria. The Wavelet Network model is introduced in section 3. A brief introduction to wavelet transform methods and Neural Networks is first given. The Wavelet Network model is then explained and then the learning algorithm is explained. In section 4 the Multiple Mother Wavelet Network is explained. In section 5 criticism of the Wavelet Network models is given. In section 6 the two GA-based models are discussed. An introduction is given to genetic algorithms. The problem of function approximation is then stated in the context of genetic algorithms and a method for it is proposed. In section 7 the above GA-based model is modified to introduce a penalty term and the corresponding model is explained. In section 8 an innovative algorithm for function forecasting is discussed which uses the idea of a *Green function* as used in Theoretical Physics. Finally the experimental results are disscussed in section 9 and the conclusion is given in section 10.

2 The Autoregressive (AR) model

In this model the current value of the process is expressed as a finite , linear aggregate of previous values of the process and a "shock" a_t.

Let us denote the values of a process at equally spaced times t,t-1,t-2, ... , by $Z_t, Z_{t-1}, Z_{t-2}, \ldots$, respectively.Also let $\tilde{Z}_t, \tilde{Z}_{t-1}, \tilde{Z}_{t-2}, \ldots$, be deviations from the mean μ . For example :

$$\tilde{Z}_t = Z_t - \mu. \tag{1}$$

Then,

$$\tilde{Z}_t = \phi_1 \tilde{Z}_{t-1} + \phi_2 \tilde{Z}_{t-2} + \cdots + \phi_p \tilde{Z}_{t-p} + a_t \tag{2}$$

is called an Autoregressive (AR) process of order p.

The variable Z is regressed on previous values of itself.Hence it is called an Autoregressive model.

We shall employ extensively the Backward Shift Operator B defined by

$$BZ_t = Z_{t-1} \implies B^m Z_t = Z_{t-m}. \tag{3}$$

The inverse operation is performed by the Forward Shift Operator

$$F = B^{-1}. \tag{4}$$

Another important operator is the Backward Difference Operator δ. It can be written in terms of B as

$$\delta Z_t = Z_t - Z_{t-1} = (1-B)Z_t \tag{5}$$

$$\implies \delta = (1-B). \tag{6}$$

Thus coming back to our AR model we have

$$\tilde{Z}_t = \phi_1 B\tilde{Z}_t + \phi_2 B^2 \tilde{Z}_t + \cdots + \phi_p B^p \tilde{Z}_t + a_t. \tag{7}$$

This again can be written as

$$(1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p)\tilde{Z}_t = a_t. \tag{8}$$

Or economically as

$$\Phi(B)\tilde{Z}_t = a_t. \tag{9}$$

This model contains p+2 unknown parameters $\mu, \phi_1, \phi_2, \ldots, \phi_p, \sigma_a^2$ which in practice have to be estimated from the data.The additional parameter σ_a^2 is the variance of the white noise process a_t. Note that inverting (9) we can write as

$$\tilde{Z}_t = \Psi(B)a_t. \tag{10}$$

Where,

$$\Psi(B) = \Phi^{-1}(B). \tag{11}$$

Autoregressive processes can be such that their characteristics do not change throughout the time series. Loosely speaking these type of processes can be called "stationary". However Autoregressive processes can also be "nonstationary" in nature. For the process to be stationary,the ϕ's must be chosen so that the weights $\psi_1, \psi_2, \ldots$ in $\Psi(B) = \Phi^{-1}(B)$ form a convergent series. The necessary requirement for stationarity is that the Autoregressive operator

$$\Phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p, \tag{12}$$

considered as a polynomial in B of order p must have all roots of $\Phi(B) = 0$ greater than 1 in absolute value;i.e : all roots must lie outside the unit circle.

2.1 Estimation of the parameters of the AR process

There are various methods for estimating the parameters of a purely autoregressive. They are :

- Ordinary Least Squares (OLS) method of estimation may be obtained by solving a system of linear equations of the form of the standard linear regression model normal equations.

- Method of obtaining Approximate Least Squares Estimates using Yule-Walker equations. These estimates are useful at the identification stage , but can differ appreciably from the estimates obtained by Maximum Likelihood Methods in some situations.

2.2 Ordinary Least Squares Estimates (OLS) of the AR Coefficients

We assume a zero-mean autoregressive process of order p. We are given a set of N+1 observation points. To estimate the parameters $\phi_1, \dots, \phi_p$ we need to solve (according to (2)) N+1 equations of the form

$$\begin{aligned}\hat{Z}_t &= \phi_1 Z_{t-1} + \phi_2 Z_{t-2} + \cdots + \phi_p Z_{t-p} \\ \hat{Z}_{t+1} &= \phi_1 Z_t + \phi_2 Z_{t-1} + \cdots + \phi_p Z_{t-p+1} \\ &\vdots \quad \vdots \\ \hat{Z}_{t+N} &= \phi_1 Z_{t+N-1} + \phi_2 Z_{t+N-2} + \cdots + \phi_p Z_{t+N-p}\end{aligned} \tag{13}$$

The above equations can be written in the form of matrix equations as

$$\hat{\mathbf{Z}}_{\mathbf{N+1x1}} = \mathbf{Z}_{\mathbf{N+1xp}} \mathbf{\Phi}_{\mathbf{px1}} \tag{14}$$

where,

$$\hat{\mathbf{Z}}_{\mathbf{N+1x1}} = \begin{pmatrix} \hat{Z}_t \\ \hat{Z}_{t+1} \\ \vdots \\ \hat{Z}_{t+N} \end{pmatrix} \tag{15}$$

$$\hat{\mathbf{Z}}_{\mathbf{N+1xp}} = \begin{pmatrix} Z_{t-1} & Z_{t-2} & \dots & Z_{t-p} \\ Z_t & Z_{t-1} & \dots & Z_{t-p+1} \\ \vdots & & \dots & \vdots \\ Z_{t+N-1} & Z_{t+N-2} & \dots & Z_{t+N-p} \end{pmatrix} \tag{16}$$

and,

$$\mathbf{\Phi}_{\mathbf{px1}} = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_p \end{pmatrix} \tag{17}$$

We wish to minimise the mean squared error

$$e^2 = \frac{1}{N+1}\sum_{t=1}^{N+1}[Z_t - \hat{Z}_t]^2 \tag{18}$$

If we differentiate e^2 w.r.t the parameters ϕ_i we get an estimate of $\phi_1, \ldots, \phi_p$ which can be written in terms of the above matrices as

$$\mathbf{\Phi_{px1}} = (\mathbf{Z'_{pxN+1}Z_{N+1xp}})^{-1}\mathbf{Z'_{pxN+1}Z_{N+1x1}} \tag{19}$$

The above equation gives the Ordinary Least Squares estimate of the parameters of an AR process. The main problem with estimation using OLS estimates is that it fails miserably in the presence of noise in the data. OLS estimates are highly susceptible to noise.

2.3 Yule-Walker Estimates

A common method of obtaining estimates $\hat{\phi}_j$ of ϕ_j is by means of solving the Yule-Walker equations.The Yule-Walker equations are as follows :

Let the autocovariances be calculated as,

$$\hat{\rho}(k) = \frac{1}{N}\sum_{t=1}^{N-|k|}(Z_{t+k} - \mu)(Z_t - \mu). \tag{20}$$

Where,

$$\mu = \frac{1}{N}\sum_{t=1}^{N} Z_t \tag{21}$$

There are of course, other methods for estimating the autocovariances , but the estimator (20) has the attractive property that its mean squared error is generally smaller than that of other estimators [Jenk'69]. The Yule-Walker equations can then be written as :

$$R\Phi = r \tag{22}$$

where,

$$R = \begin{pmatrix} \hat{\rho}(0) & \hat{\rho}(1) & \ldots & \hat{\rho}(p-1) \\ \hat{\rho}(1) & \hat{\rho}(0) & \ldots & \hat{\rho}(p-2) \\ \vdots & & \ldots & \vdots \\ \hat{\rho}(p-1) & \hat{\rho}(p-2) & \ldots & \hat{\rho}(0) \end{pmatrix} \tag{23}$$

and,

$$r = \begin{pmatrix} \hat{\rho}(1) \\ \hat{\rho}(2) \\ \vdots \\ \hat{\rho}(p) \end{pmatrix} \tag{24}$$

Equation 22 can then be inverted to get the AR coefficients $\phi_1, \ldots, \phi_p$. In particular,the estimates for first- and second-order autoregressive processes are

$$AR(1)\ \hat{\phi}_1 = r_1 \tag{25}$$

$$AR(2)\ \hat{\phi}_1 = \frac{r_1(1-r_2)}{1-r_1^2} \quad \hat{\phi}_2 = \frac{r_2 - r_1^2}{1-r_1^2} \tag{26}$$

2.4 Model Selection Criteria

In the previous sections it was assumed that the order p of the AR process was known. However the value of p may be very different for different processes and optimal values have to be found out for each process. If the value of the order , p used in the model is less than the optimal value of p for the corresponding time series then the model is incomplete and will give erroneous prediction. If the value of p used is larger than the optimal value then theoretically the extra coefficients should be zero (or at most negligible). However in practical cases they result in an overhead in computation time. The method of determining the order of an AR process is called model selection.

An approach to model selection is based on the use of certain *information criteria* such as the *Akaike Information Criterion* (AIC) proposed by Akaike [Akai'74] and the *Bayesian Information Criterion* (BIC) proposed by Schwarz [Schw'78]. In the implementation of this approach , a range of potential AR models is estimated by Maximum Likelihood methods,and for each a criterion such as the AIC (normalised by sample size n) given by :

$$AIC_p = \frac{-2ln(Max.Likelihood) + 2r}{n} \simeq ln(\hat{\sigma}_a^2) + \frac{2r}{n} + const. \tag{27}$$

$$BIC = ln(\hat{\sigma}_a^2) + \frac{rln(n)}{n}. \tag{28}$$

is evaluated,where $\hat{\sigma}_a^2$ denotes the Maximum Likelihood Estimate of σ_a^2,and $r = p+1$ denotes the number of parameters estimated in the model, including a constant term.

In the AIC criterion above,the first term essentially corresponds to $-\frac{2}{N}$ times the natural log of the maximum likelihood while the second term is a penalty factor for the inclusion of additional parameters in the model. The constant term in the approximate expression for the AIC varies according to various interpretations of the Akaike Information Criterion . The model corresponding to the minimum value of the AIC is taken as the most suitable model for representing the time series.

One immediate disadvantage of this approach is that several models have to be estimated which is computationally very time consuming and expensive. Hannan and Rissanen [Hann'82] proposed another model selection procedure.

At the first stage of the procedure,one obtains estimates of the random "shock" series a_t,by approximation of the unknown AR model by a (sufficiently high order) AR model of order m^*. The order m^* of the approximating AR model might itself be chosen by use of the AIC (or BIC) selection criterion above.From the AR(m^*) model selected one obtains residuals :

$$\tilde{a}_t = \tilde{w}_t - \sum_{j=1}^{m^*} \hat{\phi}_{m^* j} \tilde{w}_{t-j}$$

In the second stage of the procedure,one regresses $\tilde{w}_t$ on $\tilde{w}_{t-1}, \dots, \tilde{w}_{t-p}$ and $\tilde{a}_{t-1}, \dots, \tilde{a}_{t-q}$ for various values of p , i.e., one estimates approximate models of the form :

$$\tilde{w}_t = \sum_{j=1}^{p} \phi_j \tilde{w}_{t-j} + a_t \qquad (29)$$

by Ordinary Least Squares regression , and let $\hat{\sigma}_p^2$ denote the estimated error variance(uncorrected for degrees of freedom).Then by application of the BIC (or the AIC) above , the order p of the AR model is chosen as the one that minimises $ln(\hat{\sigma}_p^2) + p\frac{ln(n)}{n}$.

The appeal of this procedure is that computation of Maximum Likelihood Estimates over a wide range of ARMA models is avoided. Such procedures and use of information criteria in model selection have been useful but they should be viewed as supplementary guidelines to assist in the model selection process.In particular they should not be used as a substitute for careful examination of characteristics of the estimated autocorrelation and partial autocorrelation functions of the series. Critical examination of the residuals $a_t, \tilde{a}_t$ for model inadequacies should always be included as a major aspect of the overall model selection process.

3 The Wavelet Network Model

This is one of the recent methods for function approximation which utilises the adaptive nature of Neural Networks on one hand and the principle of decomposition of Wavelet Transforms on the other hand.In this section first the theory of wavelet transforms will be introduced. Then a brief description of Feedforward Neural

Networks will be given.Finally it will be shown how these two tools can be integrated into a single system for the purpose of function approximation and forecasting.

3.1 Wavelet Transforms

Wavelet theory is a unification of similar ideas from different fields. It is built on a strong mathematical foundation based on the idea of looking at signals at different scales and resolutions . We consider the Continuous Wavelet Transform (CWT) and the Discrete Wavelet Transform (DWT) separately .

3.1.1 Continuous Wavelet Transform

In the continuous case the wavelet tarnsform is constructed from a single function $\psi(x)$ called the *mother wavelet.*

$$\psi_{\sigma,\tau}(x) = \frac{1}{\sqrt{\sigma}}\psi(\frac{x-\tau}{\sigma}) \qquad (30)$$

where ,

σ is a scale , or dilation , factor ,

τ is the translation ,
$\frac{1}{\sqrt{\sigma}}$ is a constant that is used for *energy* normalisation.
The definition of the continuous wavelet transform (CWT) w.r.t a particular function $f(x)$ is then

$$W(\tau,\sigma) = \frac{1}{\sqrt{\sigma}}\int f(x)\psi(\frac{x-\tau}{\sigma})dx \qquad (31)$$

The wavelet transform can be seen as projecting the function $f(x)$ onto a set of basis functions , $\{\psi_{\sigma,\tau}\}$. From a mathematical point of view it would be preferable to have an orthogonal basis to minimise redundancy and interference between different wavelets. This assures that it is possible to reconstruct the function from its transform by summing up inner products as

$$f(x) = \int_{\sigma>0} W(\tau,\sigma)\psi_{\sigma,\tau}(x)\frac{d\tau d\tau}{\sigma^2} \qquad (32)$$

Surprisingly enough this holds even though the wavelets $\psi_{\sigma,\tau}(x)$ are not orthogonal [Daub'90]. The requirements on the wavelets are :

1. $\psi(x)$ should be of finite energy , i.e.

$$\int |\psi(x)|^2 dx < \infty,$$

2. $\psi(x)$ should be of the bandpass type , meaning that the reconstruction scheme in (32) is only for the function $f(x)$, the DC-component cannot be reconstructed. This means that $\psi(x)$ are all zero mean functions.

3.1.2 Discrete Wavelet Transforms

A natural question to ask is if it is possible to find an orthogonal wavelet basis by a careful sampling of ψ and τ. The answer is that it is possible but it depends critically on the choice of $\psi(x)$. There is a trade-off between the orthogonality and the restrictions on the wavelet. If the redundancy is high , there are mild restrictions on $\psi(x)$. But if the redundancy is small then the choice of the mother wavelet is very constrained [Daub'90].

The coefficients $w_{i,j}$ in the discrete wavelet transform (DWT) are given by

$$w_{i,j} = \frac{1}{\sqrt{s_j}} \int f(x)\psi(\frac{x - t_{i,j}}{s_j})dx = < \psi_{i,j}, f > \tag{33}$$

where ,

$$\psi_{i,j} = \psi(\frac{x - t_{i,j}}{s_j})$$

and the s_j and $t_{i,j}$ are the scaling and translation parameters respectively. The inverse transform in terms of the coefficients $w_{i,j}$ is given by

$$f(x) = \sum_{i=0}^{\infty}\sum_{j=1}^{\infty} w_{i,j}\psi(\frac{x - t_{i,j}}{s_j}) \tag{34}$$

To analyse the behavior of the DWT , the notation *frame* from Daubechies [Daub'90] is an appropriate concept. The frame bounds, c_{min} and c_{max} of the wavelet transform are defined according to

$$c_{min}\|f\|^2 \leq \sum_{i,j} \| < \psi_{i,j}, f > \|^2 = \sum_{i,j} \|w_{i,j}\|^2 \leq c_{max}\|f\|^2 \tag{35}$$

independent of all functions $f \in \mathcal{L}^2$. Then $\psi_{i,j}$ is a frame if $c_{min} > 0$ and $c_{max} < \infty$.

These frame bounds balance the demand on the wavelet. If all the $\psi_{i,j}$ are normalised the following terminology is used.

$$c_{min} = c_{max} = 1 \Longrightarrow Orthonormal\, basis.$$

$$c_{min} = c_{max} > 1 \Longrightarrow Tight\, frame,\, the\, number\, c_{min}\, is\, a\, measure\, of\, the\, redundancy.$$

$$c_{min} \simeq c_{max} \implies \mathit{Snug\ frame,\ the\ inverse\ is\ well-behaved.}$$

3.2 Wavelet Decompositions as Universal Approximants

In this section we will discuss how wavelets can be used for approximating any function whose functional form is unknown. We are interested in finding some appropriate function

$$\psi : \Re^n \rightarrow \Re$$

with the following property :
There exists a denumerable family of the form :

$$\Phi = \{det(D^{1/2})\psi[D_k x - t_k] : t_k \in \Re^n, D_k = diag(d_k), d_k \in \Re^n_+, k \in \mathcal{Z}\} \tag{36}$$

(where the t_k's are translation vectors and the d_k's are dilation vectors specifying the diagonal dilation matrices D_k) satisfying the frame property :
There exists two constants $c_{min} > 0$ and $c_{max} < \infty$ such that, for all f in $\mathcal{L}^2(\Re^n)$ the following inequalities hold :

$$c_{min}\|f\|^2 \leq \sum_{\psi \in \Phi} |< \psi, f >|^2 \leq c_{max}\|f\|^2 \tag{37}$$

In this sum $< . , . >$ denotes the inner product in $\mathcal{L}^2(\Re^n)$ and the sum ranges over all elements of the family Φ.
The diag() refers to an arrangement where the elemnts of an n-dimensional vector are arranged as the diagonal elements of a square matrix of order n.Hence here the D_k's and the d_k's are related as :

$$D_k = \begin{pmatrix} d_k^1 & 0 & \dots & 0 \\ 0 & d_k^2 & \dots & 0 \\ \mathbf{0} & & \ddots & \mathbf{0} \\ 0 & 0 & \dots & d_k^n \end{pmatrix} \tag{38}$$

where $d_k^1, \dots, d_k^n$ are the components of the n-dimensional vectors d_k.

$$d_k = \begin{pmatrix} d_k^1 \\ d_k^2 \\ \vdots \\ d_k^n \end{pmatrix}$$

It is a consequence of (37) that the family Φ is dense in $\mathcal{L}^2(\Re^n)$,and the quality of the best approximation for fixed number of terms in the sum depends on how far from 1 the coefficients c_{min} and c_{max} are.Hence the collection of all linear combinations of elements of the frame Φ of the form :

$$g(x) = \sum_{k=1}^{N} w_k \phi_k(x), \quad \phi_k \in \Phi, \tag{39}$$

is dense in $\mathcal{L}^2(\Re^n)$.

From this it also follows that the collection of all finite sums of the form :

$$g(x) = \sum_{i=1}^{N} w_i det(D_i)^{1/2} \psi[D_i x - t_i] \tag{40}$$

where the t_i's are arbitrary translation vectors and the d_i's are arbitrary dilation vectors specifying the diagonal dilation matrices D_i is also dense in $\mathcal{L}^2(\Re^n)$Since it contains in particular all finite linear combinations of elements of the frame Φ.There are obviously more degrees of freedom in class (40) than in class (39) since translations and dilations are not constrained to belong to the denumerable families specified in (36). This additional flexibility can be used to fit those dilations and translations to a particular function.

It remains to exhibit families Φ as above.The Wavelet theory will be used for this purpose.

Using the Continuous Wavelet Decomposition : The continuous wavelet decomposition allows us to decompose any function f(x) $\in \mathcal{L}^2(\Re^n)$ using a family of functions obtained by dilating and translating a single wavelet function :

$$\psi : \Re^n \rightarrow \Re \tag{41}$$

To build such a wavelet we proceed as follows.Consider first a scalar wavelet in the Morlet-Grossman sense [Daub'90], i.e., a function

$$\psi_s : \Re \rightarrow \Re \tag{42}$$

with a Fourier transform $\hat{\psi}_s(\omega)$ satisfying the condition

$$C_{\psi_s} = \int_0^\infty \frac{| \hat{\psi}_s(\omega) |^2}{(\omega)} d\omega < \infty. \tag{43}$$

Using this scalar wavelet ψ_s we build the desired wavelet $\psi : \Re^n \rightarrow \Re$ by setting

$$\psi(x) = \psi_s(x_1) \ldots \psi_s(x_n) \; for \; x = (x_1, x_2, \ldots, x_n) \tag{44}$$

and we also introduce

$$C_\psi = C^n_{\psi_s}.$$

Then the following continuous wavelet decomposition formulae hold :

$$f(x) = \frac{1}{C_\psi} \int_{\Re^n} \int_{\Re^n_+} W(d,t)(det\, D)^{1/2} \psi[D(x-t)] dd t \tag{45}$$

$$W(d,t) = \int_{\Re^n} f(x)(det\, D)^{1/2} \psi[D(x-t)] dx \quad where, as before \quad D = diag(d) \tag{46}$$

In these formulas d and t are the dilation and translation vectors,respectively.This result is classical for one dimension and easily extends to the multidimensional case.It is also shown in [Daub'90] that the denumerable family :

$$\Psi_s(\alpha, \beta) = \{\alpha^{k/2} \psi_s(\alpha^k x - \beta l) : k, l \in \mathcal{Z}\} \tag{47}$$

for commonly used wavelets such as the "Morlet" wavelet for instance, yield c_{min} and c_{max} very close to 1 for $\alpha = 2, \beta = 1$.

Finally there is even no need that the *synthesis* and *decomposition* formulae ((45) and (46),respectively) use the same wavelet ψ . Different wavelets in duality can be used as well. To summarise,the continuous wavelet transform theory in the Morlet-Grossman sense provides with considerable flexibility in designing our networks.

Referring to our objectives we may state the following at this point :

• The universal approximation property is guaranteed for finite sums of the form (40) if ψ is chosen as an appropriate wavelet according to the discussion above.

3.3 Neural Networks

Fig. 1 depicts a so-called (1 + 1/2) layer Neural Network. Recently the ability of such Neural Networks to approximate continuous functions has been widely studied [HecN '89]. In particular,the following result has been proved.

If $\sigma(.)$ is a continuous discriminatory function, then finite sums of the form :

$$g(x) = \sum_{i=1}^{N} w_i \sigma(a_i^T x + b_i) \tag{48}$$

are dense in the space of continuous functions defined on $[0,1]^n$ and any $\epsilon > 0$, there is a sum g(x) of the form (48), for which $\mid g(x) - f(x) \mid < \epsilon$ for all $x \in [0,1]^n$.

Any bounded and measurable sigmoidal function is discriminatory. In particular,any continuous sigmoidal function is discriminatory.In the construction of the Wavelet Network we have the following objectives in mind :

• Preserve the "universal approximation property",i.e.,provide a class of networks exhibiting the same density property as the class (48).

• Have an explicit link between the network coefficients and some appropriate transform.This will be extremely useful in guessing good initial values for the backpropagation-like algorithm.

• Possibly achieve the same quality of approximation with a network of reduced size.

We shall discuss in our conclusion how far these objectives have been achieved.

3.4 Wavelet Networks and Their parametrization

Based on the previous discussion , we propose a network structure of the form :

$$g(x) = \sum_{i=1}^{N} w_i \psi[D_i(x - t_i)] + \bar{g} \tag{49}$$

where the additional (and redundant) parameter $\bar{g}$ is introduced to help deal with with nonzero mean functions on finite domains.Note that this form (49) is equivalent to the form (40) up to the constant $\bar{g}$ since the dilations and translations are adjustable.Furthermore,in order to compensate for the orientation selective nature of the dilations in (49) we combine a rotation with each affine transform to make the network more flexible. Our Wavelet Network structure is thus of the following form :

$$g(x) = \sum_{i=1}^{N} w_i \psi[D_i R_i(x - t_i)] + \bar{g} \tag{50}$$

where,

• the additional parameter $\bar{g}$ is introduced in order to make it easier to approximate functions with nonzero average,since the wavelets used are zero mean wavelets.,

• the dilation matrices D_i's are diagonal matrices built from dilation vectors,while R_i's are rotation matrices.This network structure is illustrated in Fig. 2.

3.5 Learning Algorithm for the Wavelet Network

In this subsection an algorithm for adjusting the parameters of the Wavelet Network is discussed [Zhan'92]. The learning algorithm is based on a sample of random input-output pairs {x,f(x)} where f(x) is the function to be approximated. A stochastic gradient type algorithm is used for this purpose.It is very similar to the Backpropagation algorithm used for Neural Network learning.More precisely, in the sequel we are given a sequence of random pairs $\{x_K, y_k = f(x_k) + v_k\}$ where $\{v_k\}$ is observation noise.

3.5.1 Principle of the Stochastic Gradient Algorithm

All the parameters $\bar{g}, w_i's, t_i's, D_i's$ and $R_i's$ are collected in a vector θ and $g_\theta(x)$ is used to refer to the network defined by (50) with the parameter vector θ.The objective function to be minimised is :

$$C(\theta) = \frac{1}{2}E\{[g_\theta(x) - y]^2\} \tag{51}$$

Off-line gradient or Newton methods could also have been used but the computation of the gradient or the Hessian matrix is generally very heavy.The Stochastic Gradient algorithm modifies the parameter vector θ after a set of measurements (called a pass), in the opposite direction of the gradient of the functional :

$$c(\theta, x_k, y_k) = \frac{1}{2}[g_\theta(x_k) - y_k]^2 \tag{52}$$

3.5.2 Handling the Dilation Parameters

Up to now the dilation parameters have been denoted by :

$$D_i = diag(d_i) = diag(d_i^1, \ldots, d_i^n)$$

where d_i measures the inverse of the scale of the concerned wavelet.Another possible choice is $D_i = diag(\frac{1}{s_i^1}, \ldots, \frac{1}{s_i^n})$.The related vector $s_i = (s_i^1, \ldots, s_i^n)^T$ directly measures the scale of the concerned wavelet.If we compare the partial derivatives of the functional (52) w.r.t d_i and s_i,

$$\frac{\partial c}{\partial d_i} = e_k w_i diag[R_i(x_k - t_i)]\psi'[D_i R_i(x_k - t_i)] \tag{53}$$

$$\frac{\partial c}{\partial s_i} = e_k w_i D_i^2 diag[R_i(x_k - t_i)]\psi'[D_i R_i(x_k - t_i)] \tag{54}$$

where $\psi'(x) = \frac{d\psi(x)}{dx}$ and $e_k = g_\theta(x_k) - y_k$, we observe that the partial derivatives in (54) are smaller for larger scales,i.e., larger s_i's,a nice property which is not satisfied by (53). In order to make the Stochastic Gradient algorithm more cautious for larger scale wavelets,the form (54) is preferred.

3.5.3 Calculating the Stochastic Gradient

Explicit formulae for the partial derivatives of the functional (52) w.r.t each component of the parameter vector θ are now listed. From these formulae the calculation of the gradient follows immediately. For convenience, the following notations are used :

$$\psi'(x) = \frac{d\psi(x)}{dx},$$
$$e_k = g_\theta(x_k) - y_k$$

and

$$z_i = D_i R_i (x_k - t_i)$$

.

The partial derivatives of the functional $c(\theta, x_k, y_k)$ w.r.t w_i, t_i and s_i are respectively given by :

$$\frac{\partial c}{\partial \bar{g}} = e_k \tag{55}$$

$$\frac{\partial c}{\partial w_i} = e_k \psi(z_i) \tag{56}$$

$$\frac{\partial c}{\partial t_i} = -e_k w_i R_i^T D_i \psi'(z_i) \tag{57}$$

$$\frac{\partial c}{\partial s_i} = -e_k w_i D_i^2 diag[R_i(x_k - t_i)]\psi'(z_i) \tag{58}$$

At this point we are ready to implement the Stochastic Gradient algorithm. Additional work is however required.As is the case for Backpropagation algorithms for Neural Network learning, the objective function (51) is likely to be highly convex,so local minima are expected.To improve the situation,careful initialization of the network is performed and appropriate constraints are set on the adjusted parameters.

While no theoretical investigation is available which may guarantee convergence , drastic improvement was exhibited.The unconstrained Stochastic Gradient usually diverged but the modified algorithm performed well in all the experiments. A similar such situation is also encountered in Feedforward Neural Network training. The corresponding additional features are now presented.

3.5.4 Setting Constraints on the Adjustable Parameters

Let $f : \mathcal{D} \to \Re$ be the function to be approximated, where $\mathcal{D} \subset \Re^n$ is the domain where the approximation should be made.The following constraints on the parameters are introduced :

1. To keep the wavelets inside or near to the domain $\mathcal{D}$, select another domain $\mathcal{E}$ such that $\mathcal{D} \subset \mathcal{E} \subset \Re^n$, and require :

$$t_i \in \mathcal{E}, \quad i = 1, \ldots, N \tag{59}$$

   In other words , we choose a subspace $\mathcal{E}$ of $\Re$ containing $\mathcal{D}$ in such a manner that the value of the wavelet functions are appreciable within $\mathcal{E}$ and negligible outside it. This is necessary to calculate the wavelet transform coefficients (though the forward transform equations are not used in this model).

2. To avoid excessive compression of each wavelet, select $\epsilon > 0$ and require :

$$D_i^{-1} > \epsilon I, \quad i = 1, \ldots, N \tag{60}$$

   This is necessary to keep the number of wavelets required for approximating the function properly , as low as possible.

3. To prevent the total volume of the wavelet supports from being too small,select a $V > 0$ and require :

$$\sum_{i=1}^{N} (det\, D_i)^{-1} > V \tag{61}$$

   This is necessary for the same reason as 1.

It is to be noted that the wavelets used are not necesarily compactly supported but are always rapidly vanishing,so the notion of *support* used in the third constraint should be used in an approximate snse.A formal definition for the notion of support in such a case was given by Y.C Pati and P.S Krishnaprasad in [Pati'93]. To satisfy these constraints,a *projected* stochastic gradient procedure is implemented as depicted in Fig. 3.

3.5.5 Network Initialisation

This is where the link to wavelet decomposition can be used. More precisely,an initial guess for the network parameters can be derived by using the *decomposition* formula (46) (recall that in contrast the network structure mimics tne *synthesis* formula (45). The idea is the following : using noisy input-output measurements {x,f(x)}, we can get some rough estimate of $W(h, t)$ in (46) by replacing the integration by an appropriate averaging of the observations.The initialisation procedures cited below are only simple heuristics of this idea.

1. The One-Dimensional Case : For the sake of simplicity, the one-dimensional case is considered. It is assumed that the function f(x) is to be approximated over the domain $\mathcal{D} = [a, b]$ by a network of the form :

$$g(x) = \sum_{i=1}^{N} w_i \psi(\frac{x - t_i}{s_i}) + \bar{g}.$$

It is to be noted that no rotation parameters are needed for the one-dimensional case.The initialisation of this Wavelet Network consists in evaluation of the parameters $\bar{g}, w_i, t_i$ and s_i for $i = 1, \ldots, N$. To initialise $\bar{g}$ we need to estimate the mean of the function f(x) (from its available observations) and set $\bar{g}$ to these estimated mean. The w_i's are initalised to some random numbers in the interval [0,1]. The rest of the problem is how to initialise the t_i's and the s_i's.This is done as follows :

To initialise t_1 and s_1 select a point p between a and b : $a < p < b$.The choice of this point is etailed later. Then we set :

$$t_1 = p, \quad s_1 = \zeta(b - a),$$

where $\zeta > 0$ is a properly selected constant (the value of ζ was taken as 0.5 in our experiments).The interval $[a, b]$ is then divided into to parts at the point p. In each subinterval, we recursively repeat the same procedure which will initialise t_2, s_2 and t_3, s_3 and so on,until all the wavelets are initialised.

This procedure applies in this form when the number of wavelets used is a power of 2. When this is not the case, the recursive procedure is applied as long as possible,then the remaining t_i are initialised at random for the finest scale.

2. The Multidimensional Case : For the multidimensional case,the initialisation of $\bar{g}$ and the w_i's can be performed in the same manner as for the one-dimensional case. Here $\bar{g}$ is set to be estimated as the mean of the function to be approximated , and the w_i's are initialised randomly.To initialise the t_i's and the s_i's the different coordinates are handled separately as in the one-dimensional case.Then all the possible combinations between these s_i's and t_i's can be taken.

3.6 Tentative Improvements

The selection of the number of wavelets in the network may be performed by relying on appropriate versions of standard model order criteria in Statistics, e.g., Akaike criteria or Rissanen's Minimum Description Length principle , as described in section 2.4. In all cases, such an approach amounts to adding to the objective function an additional term which penalises the number of adjusted parameters , following the principle of parsimony [BoxJ'70]. In our experiments with Wavelet Networks no *automatic* procedure was designed for adjusting the number of wavelets directly from the data.However this was implemented in an innovative *Genetic Algorithm – based Wavelet Decomposition* which will also be discussed in this article.Before that an improvised model of Wavelet Networks which uses multiple families of wavelets instead of just one, is discussed.

4 Multiple Mother Wavelet Networks

In many real-life cases the time series may not only possess multiresolution structure but also may be a composition of various functional models, for e.g.,the time series may be a superposition of monotonically increasing(or decreasing) and osscillating functional models at various levels of resolution.

In that case it is still possible to approximate the series with a "complete set" of wavelet functions belonging to a single functional family, (single mother wavelet) to any required degree of accuracy.However the number of relevant coefficients required in this case will be large and in practice will push up the computational cost considerably.

In such cases a better method would be to judiciously choose sets of wavelets belonging to several different families(several different mother wavelets). This compound set of wavelets is then used to train a similar such Wavelet Network. In such cases the relevant number of components will be much smaller than the Wavelet Network using a single mother wavelet and its family.

Such a Wavelet Network has been termed a Multiple Mother Wavelet Network. It is described by the following model :

$$g(x) = \sum_{i=1}^{N} \sum_{\alpha=1}^{No.of\,fam.} w_i^{\alpha} \psi_{\alpha}(\frac{x - t_i^{\alpha}}{s_i^{\alpha}}) + \bar{g}.$$

Here all the symbols have their usual meaning. The subscript α refers to the family (mother wavelet). The scale and translation parameters also have the subscript α

4.1 Calculating the Stochastic Gradient

Here we wish to minimise the same square error functional as (52). i.e.,

$$c(\theta, x_k, y_k) = \frac{1}{2}[g_{\theta}(x_k) - y_k]^2$$

All the results for Wavelet Networks apply to this Multiple Mother Wavelet Network model. The partial derivatives w.r.t the various parameters are calculated as follows :

The partial derivatives of the functional $c(\theta, x_k, y_k)$ w.r.t w_i, t_i and s_i are respectively given by :

$$\frac{\partial c}{\partial \bar{g}} = e_k \tag{62}$$

$$\frac{\partial c}{\partial w_i^{\alpha}} = e_k \psi_{\alpha}(z_i) \tag{63}$$

$$\frac{\partial c}{\partial t_i^\alpha} = -e_k w_i^\alpha R_i^T D_i^\alpha \psi_\alpha'(z_i) \tag{64}$$

$$\frac{\partial c}{\partial s_i^\alpha} = -e_k w_i^\alpha (D_i^\alpha)^2 diag[R_i(x_k - t_i)]\psi_\alpha'(z_i) \tag{65}$$

In the above equations the subscript α below ψ and in the w_i^α's and s_i^α's refer to the different families (different mother wavelets) of wavelets used.

At this point we are ready to implement the Stochastic Gradient algorithm for the Multiple Mother Wavelet Network. However proper initalisation is required as in the previous model. As is the case for Backpropagation algorithms for Neural Network learning, the objective function is likely to be highly convex, so local minima are expected. To improve the situation,careful initialization of the network is performed and appropriate constraints are set on the adjusted parameters.

4.1.1 Network Initialisation

The initialisation procedures used in the case of Multiple Mother Wavelet Networks is quite similar to those for the previous model (single mother) with a few variations. The initialisation procedures cited below are only simple heuristics of this idea.

1. The One-Dimensional Case : For the sake of simplicity,the one-dimensional case is considered.It is assumed that the function f(x) is to be approximated over the domain $\mathcal{D} = [a, b]$ by a network of the form :

$$g(x) = \sum_{i=1}^{N} \sum_{\alpha=1}^{No.of fam.} w_i \psi_\alpha(\frac{x - t_i^\alpha}{s_i^\alpha}) + \bar{g}.$$

   Again here,no rotation parameters are needed for the one-dimensional case.The initialisation of the Network consists in evaluation of the parameters $\bar{g}, w_i^\alpha, t_i^\alpha$ and s_i^α for $i = 1, \ldots, N$ for the different families of wavelets. To initialise $\bar{g}$ we need to estimate the mean of the function f(x) (from its available observations) and set $\bar{g}$ to this estimated mean. The w_i's are initalised to some random numbers in the interval [0,1]. The rest of the problem is how to initialise the t_i^α's and the s_i^α's. This is done as follows :

   To initialise t_1 and s_1 select a point p between a and b : $a < p < b$. The choice of this point is detailed later. Then we set :

$$t_1 = p, \quad s_1 = \zeta(b - a),$$

   where $\zeta > 0$ is a properly selected constant (the value of ζ was taken as 0.5 in our experiments).

   The interval $[a, b]$ is then divided into two parts at the point p. In each subinterval, we recursively repeat the same procedure which will initialise

t_2, s_2 and t_3, s_3 and so on, until all the wavelets are initialised. This procedure applies in this form when the number of wavelets used is a power of 2. When this is not the case, the recursive procedure is applied as long as possible, then the remaining t_i are initialised at random for the finest scale.

The above procedures for the initialisation of t_i and s_i are carried out separately for each family of wavelets.

2. The Multidimensional Case : For the multidimensional case,the initialisation of $\bar{g}$ and the w_i's can be performed in the same manner as for the one-dimensional case. Here $\bar{g}$ is set to be estimated as the mean of the function to be approximated, and the w_i^α's are initialised randomly.To initialise the t_i^α's and the s_i^α's the different coordinates are handled separately as in the one-dimensional case.Then all the possible combinations between these s_i^α's and t_i^α's can be taken.

5 Criticism of the above Wavelet Network Models

The most important criticism for Wavelet Networks comes from the fact that though it has been cited that the network coefficients have an explicit link with the actual transform coefficients of the corresponding wavelets,this can be proved to be wrong.To prove this we considered a synthetically constructed time series using the following wavelets :

$$g^1_{trans,scale}(x) = -xe^{\frac{1}{2}x^2}$$

trans. = 0.0	scale = 1.0
trans. = -5.0	scale = 5.0
trans. = 5.0	scale = 5.0

$$g^2_{trans,scale}(x) = sin(x)e^{-\frac{1}{2}x^2}$$

trans. = 0.0	scale = 1.0
trans. = -5.0	scale = 5.0
trans. = 5.0	scale = 5.0

All the coefficients w_i for the above translated and scaled wavelets were taken as 10.0 while the rest of the w_i were set to 0.

The Wavelet Network was successfully trained using 16 wavelets, 8 from the first set and 8 from the second set,including the above ones.If the above conjecture was to be true then only those coefficients w_i corresponding to the above wavelets would have had appreciably high values (~ 10.0).The forward transform for finding the wavelet transform coefficients was performed after the network was trained, using the final values of the scale and translation parameters. However we had the following results instead.

$$w^1_{0.21,1.14} = 3.928647(9.56) \quad w^2_{0.13,1.17} = 3.640312(9.89)$$

$$w^1_{-5.45,5.13} = -2.113495(9.96); \quad w^2_{5.22,5.19} = 2.302456(9.87)$$

$$w^1_{-7.56,2.35} = -1.219450(1.09) \quad w^2_{-7.16,2.33} = 4.923816(0.85)$$

$$w^1_{-2.61,2.32} = -1.109861(0.21) \quad w^2_{-2.55,2.31} = -1.671342(0.65)$$

$$w^1_{2.52,2.31} = 1.590399(1.01) \quad w^2_{2.59,2.48} = 1.490732(0.99)$$

$$w^1_{5.33,5.12} = 1.502341(9.05) \quad w^2_{5.19,5.06} = 4.156870(9.78)$$

$$w^1_{7.65,2.57} = -2.758421(1.01) \quad w^2_{7.26,2.66} = -3.498356(1.23)$$

$$w^1_{10.67,1.11} = 3.490235(0.10) \quad w^2_{10.12,1.14} = 5.121056(0.77)$$

The numbers outside the brackets denote the *network* transform coefficients while the numbers in the brackets denote the *wavelet* transform coefficients.It can be seen that the numbers in each pair differ considerably in all the cases. This is quite sufficient to show that the network coefficients have no explicit functional dependence on the wavelet transform coefficients.

6 GA-based Wavelet Decomposition

In this section an innovative *Genetic Algorithms – based* method is introduced. However before the algorithm is formally discussed, the drawbacks and of Wavelet Network models are explained and reasons are cited for the use of GA-based models for Wavelet Decomposition.

6.1 Motivation for GA-based Model

The above Wavelet Network models discussed [Zhan'92] are trained using Backpropagation - type learning algorithms using *gradient-descent* methods. The drawbacks of these methods are as follows :

• First, such methods are local in scope;the optima they seek are the best in a neighborhood of the current point.
• Second, calculus-based methods depend on the existence of derivatives (well-defined slope values). Even if we allow numerical approximation of derivatives, this is a severe shortcoming. Many practical parameter spaces have little respect for the notion of a derivative and the smoothness this applies. The real world of search is fraught with discontinuities.

It comes as no surprise that methods depending upon the restrictive requirements of continuity and derivative existence are unsuitable for all but a limited problem domain.

Enumerative schemes have been considered in many shapes and sizes. The idea is fairly straightforward;within a finite search space,or a discretized infinite search space,the search algorithm starts looking at objective function values at every point in the space,one at a time.Although the simplicity of this type of algorithm is attractive,and enumeration is much like a human kind of search (when the number of possibilities are small), such schemes must ultimately be rejected in the robustness race for one simple reason : lack of efficiency.Many practical spaces are simply too large to search sequentially and still have a chance of using the search information to some practical end as in online applications.

Even the highly touted enumerative scheme *dynamic programming* breaks down on problems of moderate size and complexity,suffering from a malady melodramatically labeled *the curse of complexity* [Bell'61]. We must conclude that less clever enumerative schemes are similarly,and more abundantly, cursed for real problems.

Random search algorithms have achieved increasing popularity as researchers have recognised the shortcomings of calculus-based and enumerative schemes.Yet, random walks and random schemes that search and save the best must also be discounted because of the efficiency requirement.Random searches,in the long run can be expected to do no better than enumerative schemes.

In our haste to discount strictly random search methods,we must be careful to distinguish them from *randomised techniques*. Genetic Algorithms are an example of a serch procedures that use random choice as a tool to guide a highly exploitative search through a coding of a parameter space. Using random choice as a tool for a directed search process seems strange at first sight but nature contains many such examples. Another currently popular search technique, *simulated annealing* uses random processes to guide its form of search for minimal energy states.The important thing to recognise at this juncture is that randomised search does not necessarily imply directionless search.

6.2 Genetic Algorithms

The mechanics of a generic genetic algorithm are surprisingly simple,involving nothing more than copying strings and swapping partial strings. The explanation of why this simple process works is much more subtle and powerful. Simplicity of operation and power of effect are two of the main attractions of the genetic algorithm approach.

A generic genetic algorithm that yields good results in many practical problems is composed of three main operations [Gold'89] :

1. Reproduction (or selection)
2. Crossover
3. Mutation

Reproduction (or selection) is a process in which individual strings are copied according to their objective function values, f (biologists call this function the fitness function). Intuitively we can think of the function f as some measure of profit,utility or goodness that we want to maximise. Copying strings according to their fitness values means that strings with a higher value have a higher probability of contributing one or more offspring in the next generation.

This operator, of course, is an artificial version of natural selection, a Darwinian method for *survival of the fittest* among string creatures.In natural populations fitness is determined by a creatures ability to survive predators, pestilence and other obstacles to adulthood and subsequent reproduction.In our unabashedly artificial settings the objective function is the final arbiter of the string creature's life or death.

The selection operator may be implemented in algorithmic form in a number of ways. Perhaps the easiest form is to create a biased *roulette wheel* where each current string in the population has a roulette wheel slot sized in proportion to its fitness.

After selection , simple crossover may proceed in two steps as follows :

• First,members of the newly generated strings in the mating pool are mated at random.
• Second, each pair of strings under crossing over as follows : an integer position k along the string is selected uniformly at random between 1 and the string length(l) less one,i.e.,$[1, l]$.Two new strings are created by swapping all characters between positions $k + 1$ and l inclusively.

Finally comes the issue of mutation.Mutation is needed because, even though reproduction and crossover effectively search and recombine extant notions,occasionally they may become overzealous and lose some potentially useful genetic material (1's or 0's at particular locations). In artificial genetic systems the mutation operator guards against such an irrecoverable loss.In the generic GA problem, mutation is the occasional (with small probability) random alterations of the value of a string position.

When used sparingly with reproduction and crossover , it is an *insurance policy* against premature loss of important information. That the mutation operator plays a secondary role in the generic GA,we simply note that the frequency of mutation to obtain good results in empirical studies is of the order of one mutation per thousand bit (position).

6.3 Difference of GAs from Traditional Methods

Genetic Algorithms are different from conventional optimization and search procedures in the following ways :

1. GA's work with a *coding* of the parameter set,not with the parameters themselves.

2. GA's scrach from a population of points not a single point as used in conventional search techniques.

3. GA's use $payoff$(objective function) information, not derivatives or other auxiliary knowledge.

4. GA's use probabilistic transition rules,not deterministic rules.

Genetic Algorithms require the natural set of parameters of the optimisation problem to be coded as a string of finite length over some finite alphabet. They exploit similarities among coded strings in a very general way. As a result they are largely unconstrained by the limitations of other methods(continuity, existence of derivatives,unimodality,and so on).

Many conventional search techniques require auxiliary information in order to work properly.For example, gradient descent techniques need derivatives (calculated analytically or numerically) in order to be able to climb the current (possibly local) peak, and other local search procedures like the greedy techniques of combinatorial optimisation can reach their goal only if they have access to most if not all auxiliary information.

By contrast, genetic algorithms have no need for all these auxiliary information: GA's form an evolutionary system. To perform an effective search for better and better structures,they only require payoff values (objective function values) associated with individual strings of a population. This characteristic makes a GA a more canonical method than many search schemes.

Unlike many methods,GA's use probabilistic transition rules to guide their search. They use random choice as a tool to guide their search towards regions of the search space with likely improvement.

Taken together, these four differences-direct use of a coding, search from a population, blindness to auxiliary information, and randomised operators -contribute to a genetic algorithm's robustness and resulting advantage over other more commonly used techniques.

However it should also be noted that not all is good with Genetic Algorithms. For non-combinatorial optimisation problems where the search space is unbounded (say the set of reals) , GA's often require prohibitively large computation time to get any reasonable solution even for a population with search space of moderate dimensions. Moreover for GA's there is also no well defined termination criterion.

6.4 Presentation of the Problem in the context of GA

In this subsection we introduce the problem of function approximation by wavelet decomposition from the context of genetic algorithms.

In the context of this problem we have several (functionally)different families of wavelets $\{g^{\alpha}_{trans,scale}\}$
$\alpha \Longrightarrow$ family number for certain values of the translation and scale parameters. Given any function f(x) which is to be approximated in some interval [a,b], we proceed as follows :

1. We first calculate the transform coefficients :

$$w^{\alpha}_{trans,scale} = < g^{\alpha}_{trans,scale}, f > \qquad (66)$$

   by performing the corresponding convolution of the function f(x) with the wavelets $g^{\alpha}_{trans,scale}(x)$ in the interval $[a,b]$, using the given values of f(x) in $[a,b]$.Note that these transform coefficients are fixed for a given function f(x) and need to be calculated only once in the beginning.

2. The coding of the wavelets in the chromosome string is an important issue.Given k families of wavelets each having m translations and n scalings, there are a total of $m.n.k$ wavelets in what may be called a *dictionary*.Then each chromosome string γ can be represented as a Boolean string as :
$\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_{mnk})$. Each bit γ_i in the string represents the presence(1) or the absence(0) of a particular wavelet from this dictionary (fixed values of family, translation and scale) weighted by the corresponding weighting factor $w^{\alpha}_{trans,scale}$. Thus for that particular chromosome (or individual) γ the approximation to f(x) is :

$$\hat{f}(x) = \sum_{\forall\gamma_i = 1} w^{\alpha_i}_{trans_i,scale_i} g^{\alpha_i}_{trans_i,scale_i}(x) \tag{67}$$

3. The next important issue is the choice of the fitness function to be maximised while choosing the best individual from a given population.The simplest and clearest choice in this aspect is the inverse of the mean square error (MSE). This is calculated as follows :

   Given a chromosome string γ,the corresponding approximation $\hat{f}(x)$ to f(x) is calculated using the formula cited above. The meam square error is then calculated as :

$$MSE = \frac{1}{N} \sum_{x \in [a,b]} [f(x) - \hat{f}(x)]^2 \tag{68}$$

   where, N = total no. of observation points in $[a, b]$.

   The inverse of this mean square error is a good measure of the fitness value for that individual.In this way the fitness values are calculated for all the individuals in a population.

6.5 Population Initialisation and Variations From the Main Theme

The starting population is initialised as follows :

1. For each individual in the population,the number of 1's(TRUE bit values) is chosen at random between 1 and $m.n.k$(chromosome length).

2. Those many allele sites are then chosen randomly in that string and the bit values are set to 1. The rest of the bit values are set to 0(FALSE).

Keeping in mind that in a pure genetic algorithmic method with no adulteration or forceful manipulation in crossover , convergence takes a very large number of generations and that too with very little guarantee , we chooose to modify this idea as follows :

In each generation we always retain the best-fit string and carry it over to the next generation if and only if its fitness value is greater than that of the best-fit string in the new generation.This guarantees that the fitness value(and hence the approximation) can only improve from generation to generation. It can never deteriorate due to sudden unwanted crossover or mutations.

7 GA-based Wavelet Decomposition with Penalty

In this section a modified genetic algorithm-based wavelet decomposition technique is proposed. This algorithm uses *penalties* together with payoffs. Considering some ofthe drawbacks and limitations of the previous technique , the need for the present modified technique can be justified easily.

Consider the case when the above GA-based method is used to approximate a function in which certain trends(particular values of the translation and scale parameters) are present with very high weightage and others are present with very low weightage.

In such a case the algorithm would, in most cases, not only select the important trends but also a large number of the unimportant ones which are not present in the original function.The reason for this is that since the unimportant trends have very low transform coefficients (w.r.t f(x)), their contribution in the best approximation (best-fit individual) to f(x) changes the mean square error and hence the fitness value by a negligible amount.Hence they may also be present in near-optimal approximations of f(x).

Clearly this is not a desirable result. The unwanted trends not only introduce some error in the approximation but also increase the computational cost since they are also evaluated every time the value of the function is calculated at some point.It would have been better if only those trends present in the original function could be retained in the best-fit individual and the rest are discarded . To implement this the penalty-based algorithm was developed. It will be discussed in the next subsection.

7.1 Function Approximation with the Modified Scheme

In this subsection we introduce the problem of function approximation by wavelet decomposition based on a genetic algorithm using penalty.

The notations and functions used here are the same as the previous genetic algorithmic approach to wavelet decomposition. Here we have several (functionally) different families of wavelets $\{g^{\alpha}_{trans,scale}\}$
$\alpha \Longrightarrow$ family number for certain values of the translation and scale parameters. Given any function f(x) which is to be approximated in some interval [a,b] , the steps of the method are described as follows :

1. We first calculate the transform coefficients :

$$w^{\alpha}_{trans,scale} = < g^{\alpha}_{trans,scale}, f > \tag{69}$$

   by performing the corresponding convolution of the function f(x) with the wavelets $g^{\alpha}_{trans,scale}(x)$ in the interval $[a,b]$,using the given values of f(x) in $[a,b]$.Note that these transform coefficients are fixed for a given function f(x) and need to be calculated only once in the beginning.

2. Next comes the coding of the wavelets in the chromosome string which is an important issue . Given k families of wavelets each having m translations and n scalings, there are a total of $m.n.k$ wavelets which may be called a *dictionary*. Now each chromosome string γ can be represented as a Boolean string as :
   $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_{mnk})$. Each bit γ_i in the string represents the presence(1) or the absence(0) of a particular strain (fixed values of family , translation and scale) of wavelet from this dictionary appropriately weighted by the corresponding weighting factor $w^{\alpha}_{trans,scale}$. Thus for that particular chromosome (or individual) γ the approximation to f(x) is :

$$\hat{f}(x) = \sum_{\forall \gamma_i = 1} w^{\alpha_i}_{trans_i,scale_i} g^{\alpha_i}_{trans_i,scale_i}(x) \tag{70}$$

3. The next important issue is the choice of the fitness function. The fitness function should be such that its highest value in a population corresponds to best-fit chromosome (or individual) in that population. We thus define the fitness function used in this problem below.

   Let n_{TRUE} be the number of alleles in the chromosome string which have values TRUE(1). We assume that the mean square error for the initial population (MSE_{init}) has been calculated and stored somewhere so that it can be used later.The objective function is defined as :

$$fitness = \frac{1}{MSE + \alpha n_{TRUE} MSE_{init}} \tag{71}$$

Here α is a positive real number, $0 < \alpha < 1$. MSE is the mean squared error. Given a chromosome string γ,the corresponding approximation $\hat{f}(x)$ to f(x) is calculated using the formula cited above. The meam square error is then calculated as :

$$MSE = \frac{1}{N} \sum_{x \in [a,b]} [f(x) - \hat{f}(x)]^2 \tag{72}$$

where, N = total no. of observation points in $[a, b]$.

Use of this objective function penalises those individuals which contain additional unnecessary wavelets.

7.1.1 Population Initialisation and Variations From the Main Theme

The starting population is initialised as follows :

1. For each individual in the population,the number of 1's(TRUE bit values) is chosen at random between 1 and mnk(chromosome length).

2. Those many allele sites are then chosen randomly in that string and the bit values are set to 1.The rest of the bit values are set to 0(FALSE).

3. The mean square error of this initial generation is then calculated using the formula given in (72) and this result is stored somewhere for future use.

Keeping in mind that in a pure genetic algorithmic method with no adulteration or forceful manipulation in crossover,convergence takes a very large number of generations and that too with very little guarantee,we divert from the original theme as follows :

In each generation we always retain the best-fit string and carry it over to the next generation if and only if its fitness value is greater than that of the best-fit string in the new generation.This guarantees that the fitness value(and hence the approximation) can only improve from generation to generation. It can never deteriorate due to sudden unwanted crossover or mutations.

8 Another Tentative Approach

This is a final tentative approach to function approximation and forecasting. It draws heavily from the *Feynman Propagator Theory* which is used in problems in Theoretical Particle Physics [Feyn'65]. In this theory the path of an elementary particle in *spacetime* is calculated by summing over all *fields* that have traversed and reached the particle at that instant of time. In effect the theory is a *causal* theory of fields and particles.

In our case let $f(x)$ be the function to be approximated, and let $\hat{f}(x)$ be an approximation to it. We use the model :

$$\hat{f}(x) = \int G(x - x')g(x)dx' \tag{73}$$

where,

$$g(x') = f(x'), \forall x' < x$$

$$= undefined\, but\, finite \forall x' \geq x. \tag{74}$$

The function G(x) is called the *Green function* or Propagator for the model. It has the causal property that

$$G(x) = 0, \forall x \leq 0. \tag{75}$$

The above equations describe the model required for the problem. It is clear that for forecasting the Green function G(x) has to be constructed from knowledge of past data only. This is done as follows. We resolve the Green function in terms of its wavelet tarnsform as

$$G(x) = \int\int W(\sigma, \tau)\psi(\frac{x - \tau}{\sigma})d\tau d\tau \tag{76}$$

where,

$$W(\sigma, \tau) = \int G(x)\psi(\frac{x - \tau}{\sigma})dx \tag{77}$$

The problem then reduces to finding the transform coefficients $W(\sigma, \tau)$. For this first we change to discrete universe.The above equations then carry over to the following equations.

$$\hat{f}(k) = \sum_{k'=-K}^{K} G(k - k')g(k') \quad K = No.\, of\, past\, obsvn. \tag{78}$$

where the functions have thier usual meaning. Then ,

$$G(k) = \sum_{i=0}^{M}\sum_{j=1}^{N} W_{i,j}\sqrt{s_j}\psi(\frac{x - t_{i,j}}{s_j}) \tag{79}$$

where the parameters have their usual meaning for discrete wavelet transform (DWT). The constants M and N are appropriately chosen. Hence substituting (79) in (78) we have

$$\hat{f}(k) = \sum_{k'=-K}^{K} \sum_{i=0}^{M} \sum_{j=1}^{N} W_{i,j} \sqrt{s_j} \psi(\frac{(k-k')-t_{i,j}}{s_j}) g(k') \tag{80}$$

The idea is to learn the wavelet transform coefficients $W_{i,j}$ instead of calculating them (which is computationally expensive) using the forward transform. This can be done by applying a Backpropagation-type learning algorithm which is described below.

8.1 Learning Algorithm

We define the mean squared error as

$$e^2 = \frac{1}{2K} \sum_{k=1}^{K} [f(k) - \hat{f}(k)]^2 \tag{81}$$

Our gradient-descent learning intends to minimise this mean squared error. The stochastic gradient for the $W_{i,j}$ are calculated as follows :

$$\frac{\partial e^2}{\partial W_{i,j}} = -\frac{1}{K} \sum_k \Delta_k [\sum_{k'} \sum_{i,j} \frac{1}{\sqrt{s_j}} \psi(\frac{(k-k')-t_{i,j}}{s_j}) g(k')] \tag{82}$$

where,

$$\Delta_k = f(k) - \hat{f}(k)$$

The coefficients are then updated in each learning pass using the generalised δ-rule :

$$W_{i,j}^{(new)} = W_{i,j}^{(old)} + \alpha \frac{\partial e^2}{\partial W_{i,j}} \tag{83}$$

where α is the learning rate.

8.2 Comments on the Above Model

The above algorithm was implemented in C language. The program was run for several iterations. However no positive result was obtained even though the error was observed to decrease at each pass. It is most likely that further improvements such as proper initialisation and proper choice of the wavelet scale and translation parameters will better results.

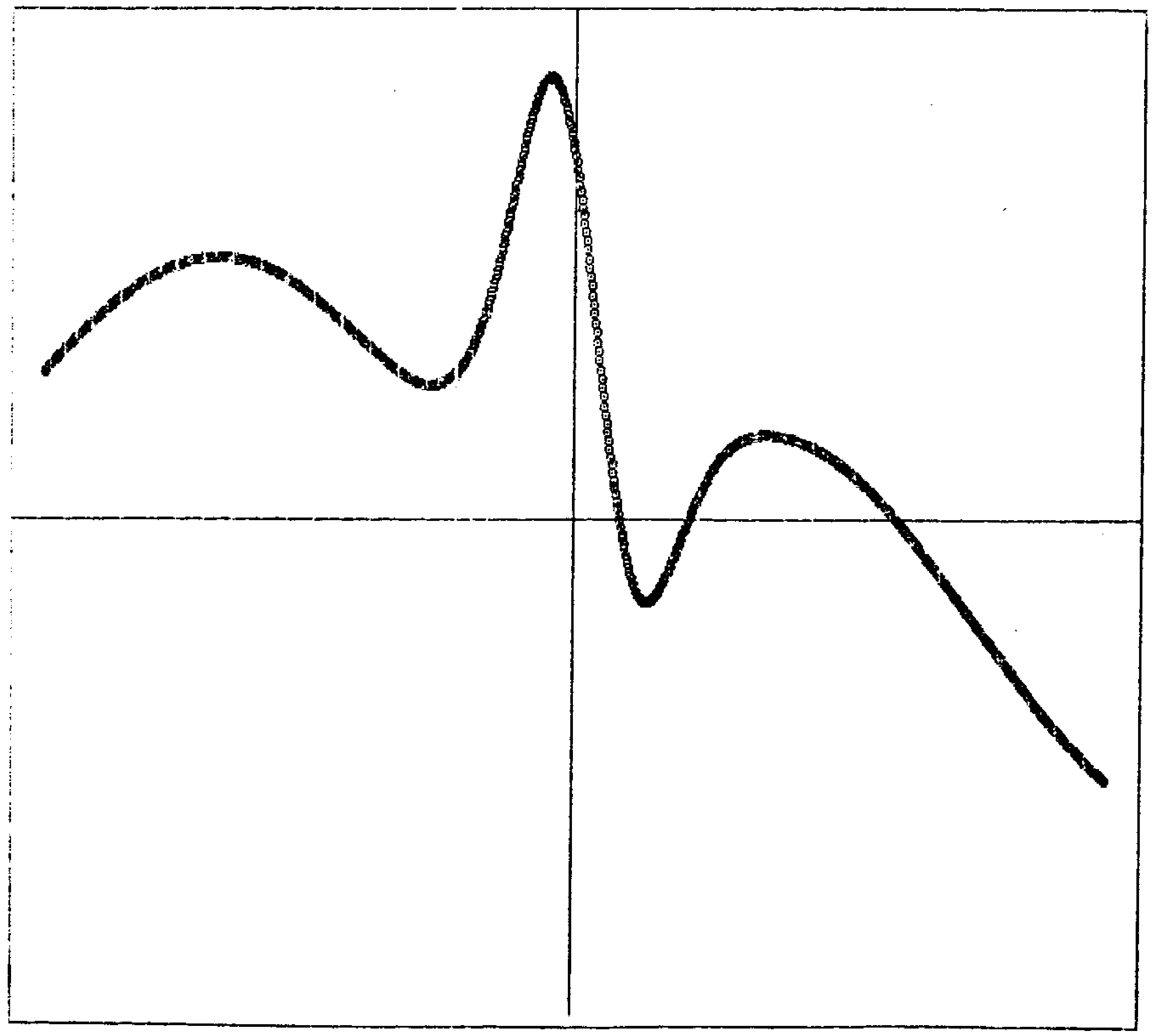


Fig. 4. (Original Graph)

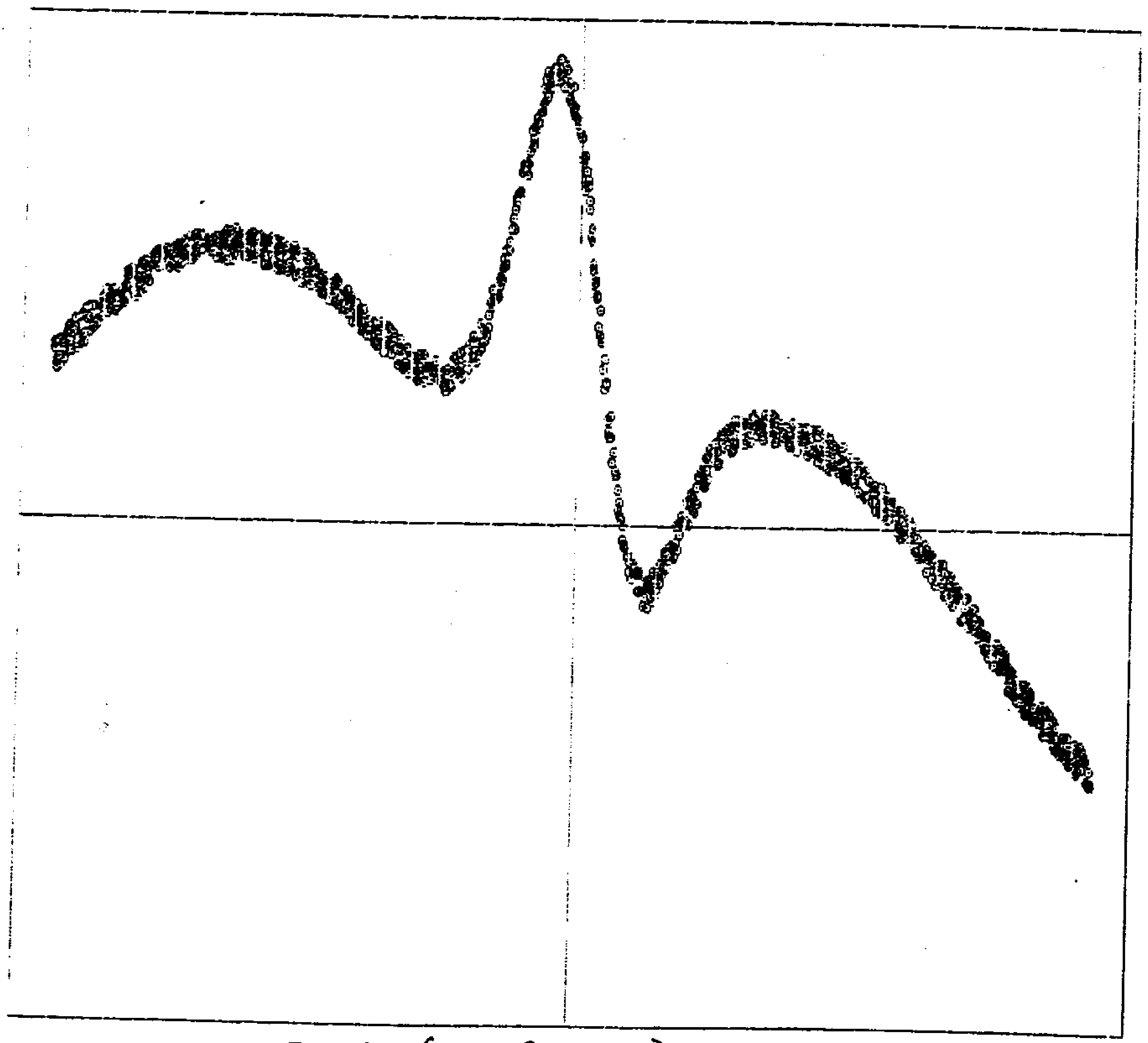


Fig. 5 (AR Process)

9 Experimental Results

In this section , results on simulated as well real data are reported. The above discussed algorithms were implemented in C language. The programs were run on Sun and Silicon Graphics workstations. The experiments were performed using the Autoregressive model , the Multiple Mother Wavelet Network model , the Unconstrained Genetic Algorithm model and the Penalty-based GA model. The original function as well as the approximations using the various models were plotted. The results are shown in the adjacent graphs.

9.1 Experiments with Simulated Data

The simulated data was prepared in the following manner. The 6 strains of wavelets (3 each from two different families of wavelets were taken to prepare the simulated data as shown.

$$g^1_{trans,scale}(x) = -xe^{-\frac{1}{2}x^2}$$

trans. = 0.0	scale = 1.0
trans. = -5.0	scale = 5.0
trans. = 5.0	scale = 5.0

$$g^2_{trans,scale}(x) = sin(x)e^{\frac{1}{2}x^2}$$

trans. = 0.0	scale = 1.0
trans. = -5.0	scale = 5.0
trans. = 5.0	scale = 5.0

All the coefficients w_i for the above translated and scaled wavelets were taken as 10.0 while the rest of the w_i were set to 0 . The actual function was plotted and is shown in Fig. 4.

For the AR process the best results (minimum error) was achieved with a 5_{th} order process. The approximated function was plotted and the result appears in Fig. 5.The AR coefficients are given below :

In the Multiple Mother Wavelet Network the network was trained with 16 wavelets taking 8 each from two different families whose functional forms are given below :

Family 1:
$g^1_{trans,scale}(x) = -xe^{-\frac{1}{2}x^2}$

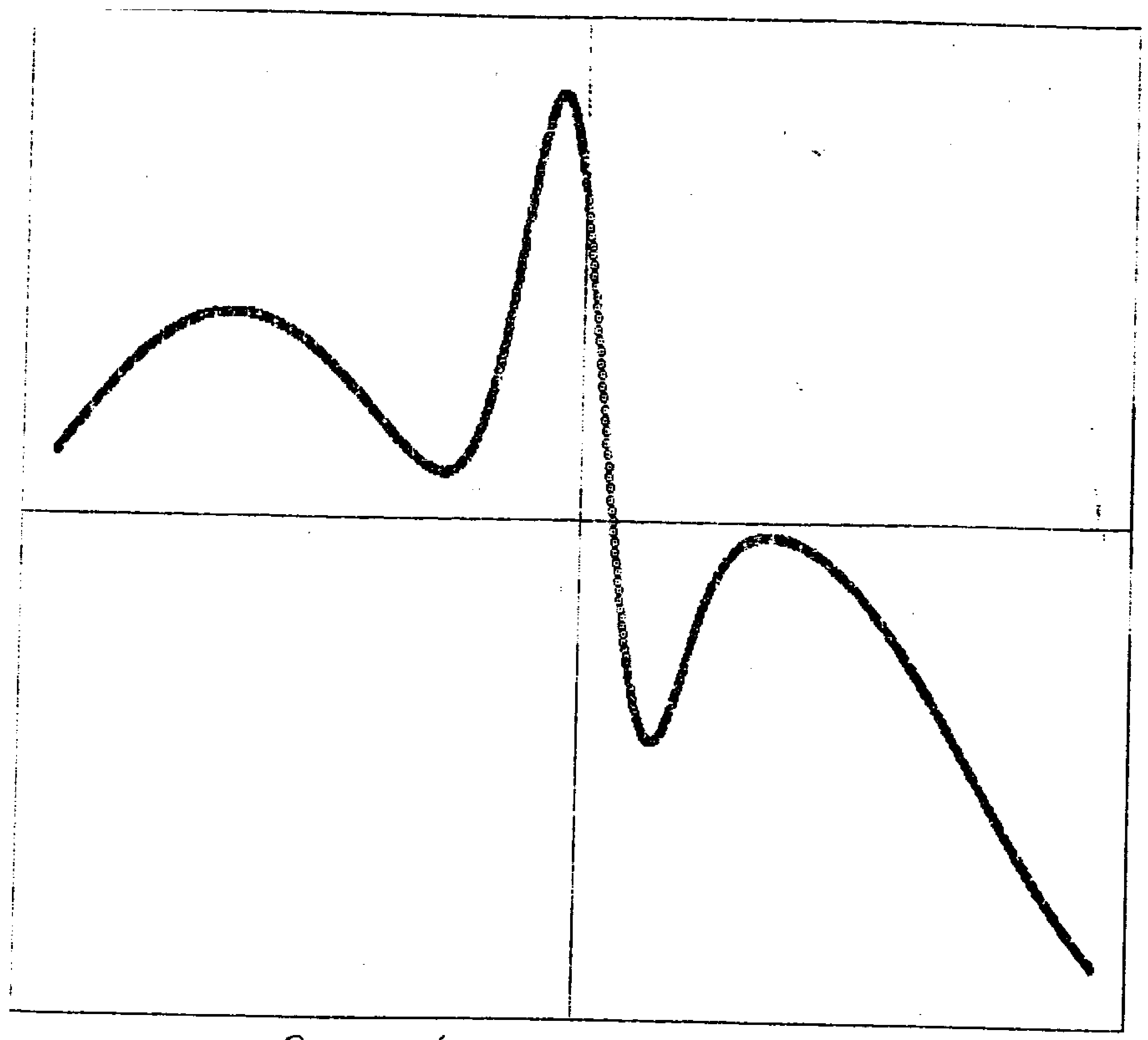


Fig. 6 (Wavelet Network)

Family 2:
$g^2_{trans,scale}(x) = sin(x)e^{-\frac{1}{2}x^2}$

The values of the translation and the scale parameters included those values that were present in the original function.The approximated function was plotted and the result is shown in Fig. 6. The values of the various transform coefficients are given below.

$$w^1_{0.21,10.14} = 3.928647 \quad w^2_{0.13,10.17} = 3.640312$$

$$w^1_{-5.45,5.13} = -2.113495; \quad w^2_{-5.22,5.19} = 2.302456$$

$$w^1_{-7.56,2.35} = -1.219450 \quad w^2_{-7.16,2.33} = 4.923816$$

$$w^1_{-2.61,2.32} = -1.109861 \quad w^2_{-2.55,2.31} = -1.671342$$

$$w^1_{2.52,2.31} = 1.590399 \quad w^2_{2.59,2.48} = 1.490732$$

$$w^1_{5.33,5.12} = 1.502341 \quad w^2_{5.19,5.06} = 4.156870$$

$$w^1_{7.65,2.57} = -2.758421 \quad w^2_{7.26,2.66} = -3.498356$$

$$w^1_{10.67,1.11} = 3.490235 \quad w^2_{10.12,1.14} = 5.121056$$

fig. 7. (Unconstrained GA)

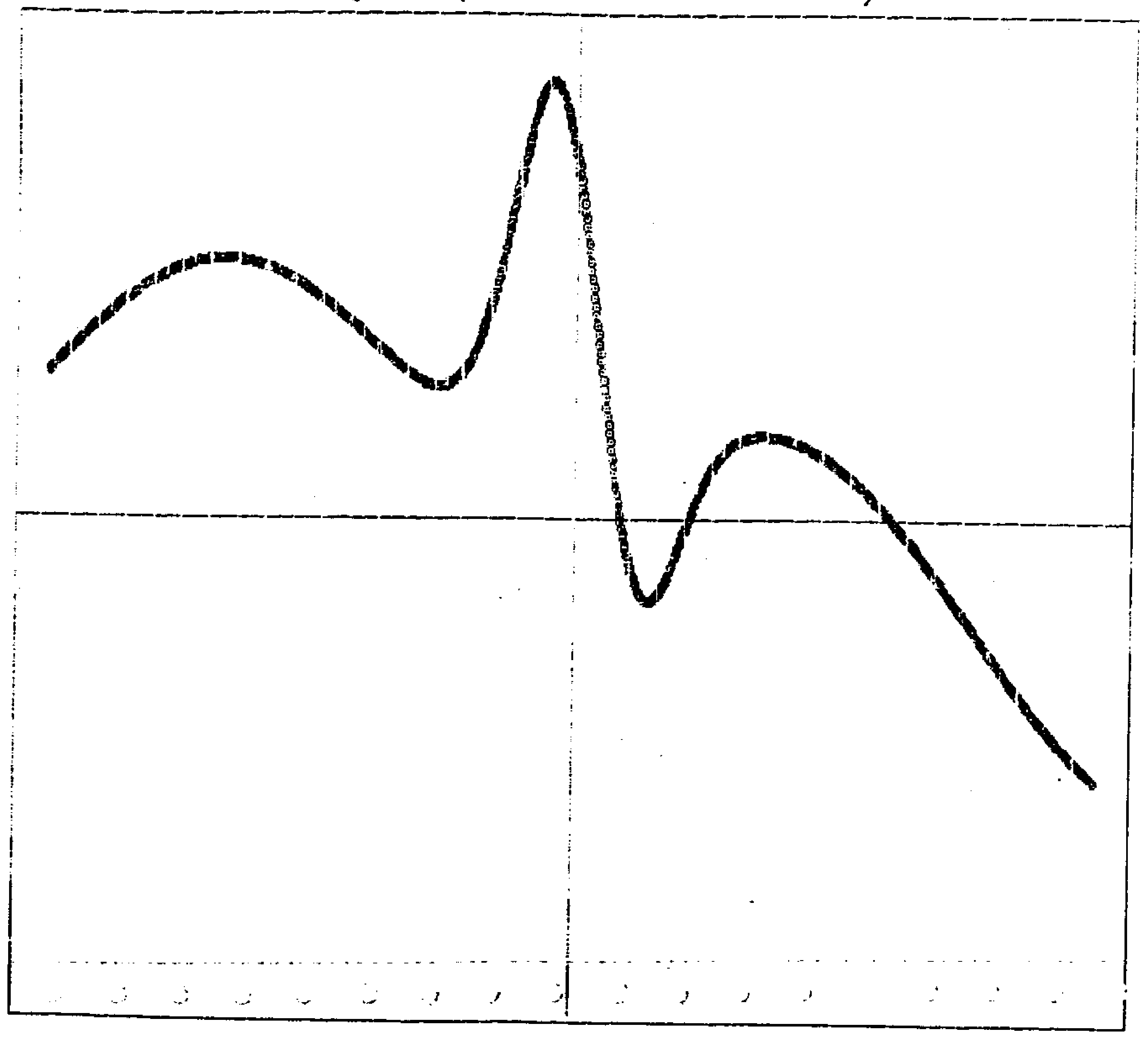
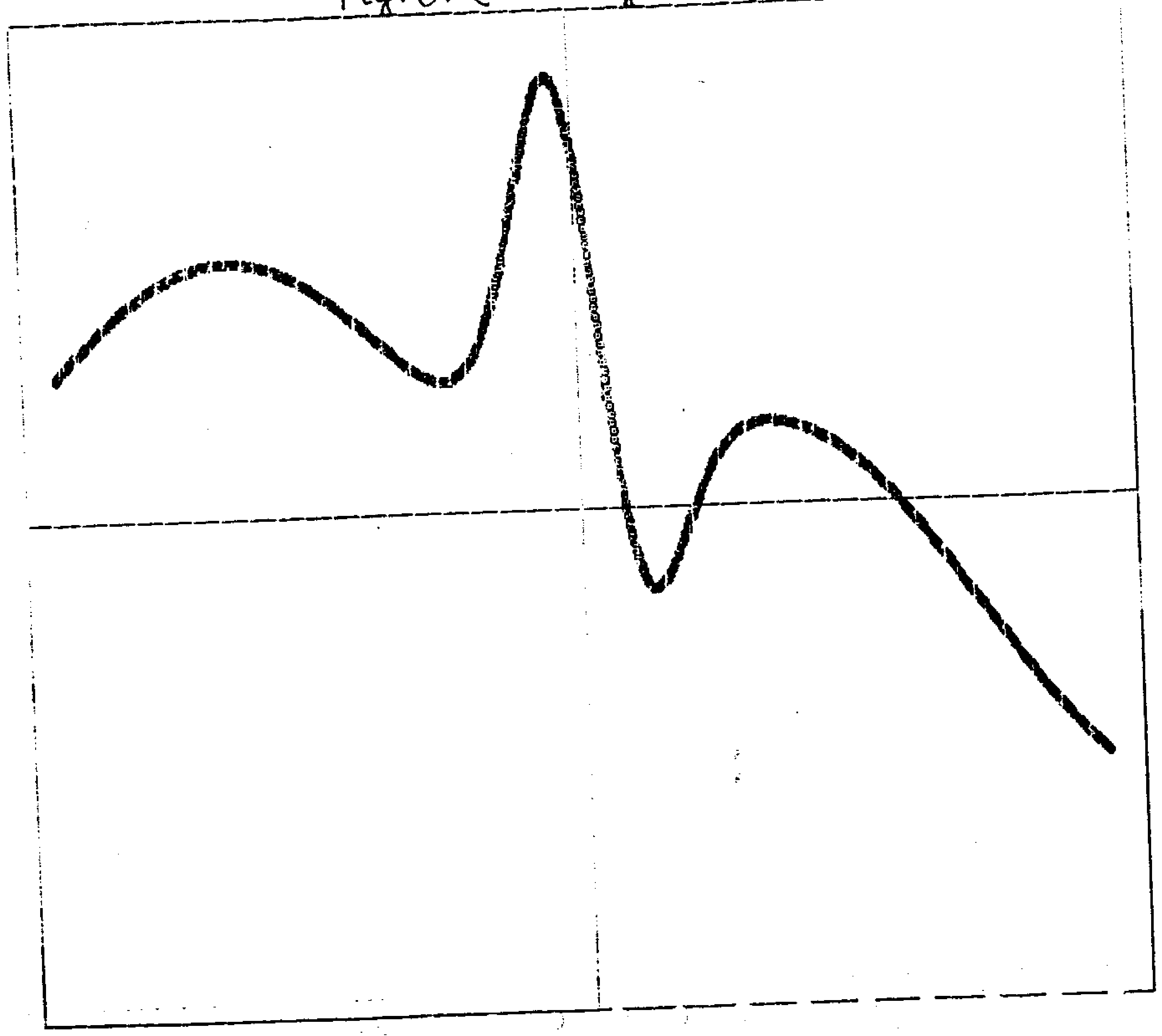


Fig. 8. (Penalty-based GA)



In the case of both the Unconstrained GA-based algorithm and the Penalty-based algorithm the *dictionary* included a total of 45 wavelets taking 15 each from 3 different functional families of wavelets.The functional forms for these families are given below :

Family 1:

$$g^1_{trans,scale}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \end{cases}$$

Family 2:

$$g^2_{trans,scale}(x) = -xe^{-\frac{1}{2}x^2}$$

Family 3:

$$g^3_{trans,scale}(x) = sin(x)e^{-\frac{1}{2}x^2}$$

The values of the translation and the scale parameters included those values that were present in the original function.The algorithm was run for 100 generations.The approximated functions for both the cases were plotted and the results are shown in Figs. 7 and 8. The best fit strings and the maximum, minimum and average fitness values for both the cases(unconstrained and penalty-based) are given below.

For the unconstrained GA method :
Maximum fitness : 46.493467
Minimum fitness : 0.010583
Average fitness : 3.539036
Best fit string : 111000010100101111100001110001111101011101000

For the penalty-based GA method :
Maximum fitness : 22.221481
Minimum fitness : 0.011082
Average fitness : 2.015031
Best fit string : 000100100000001110000001100010111101001100000

Note the marked difference in the number of 1's between the two cases. The fitness values are also different due to the penalty term used. The relevant strains are those occurring in the $16^{th}, 17^{th}, 24^{th}, 31^{st}, 32^{nd}$ and 39^{th} bit positions.

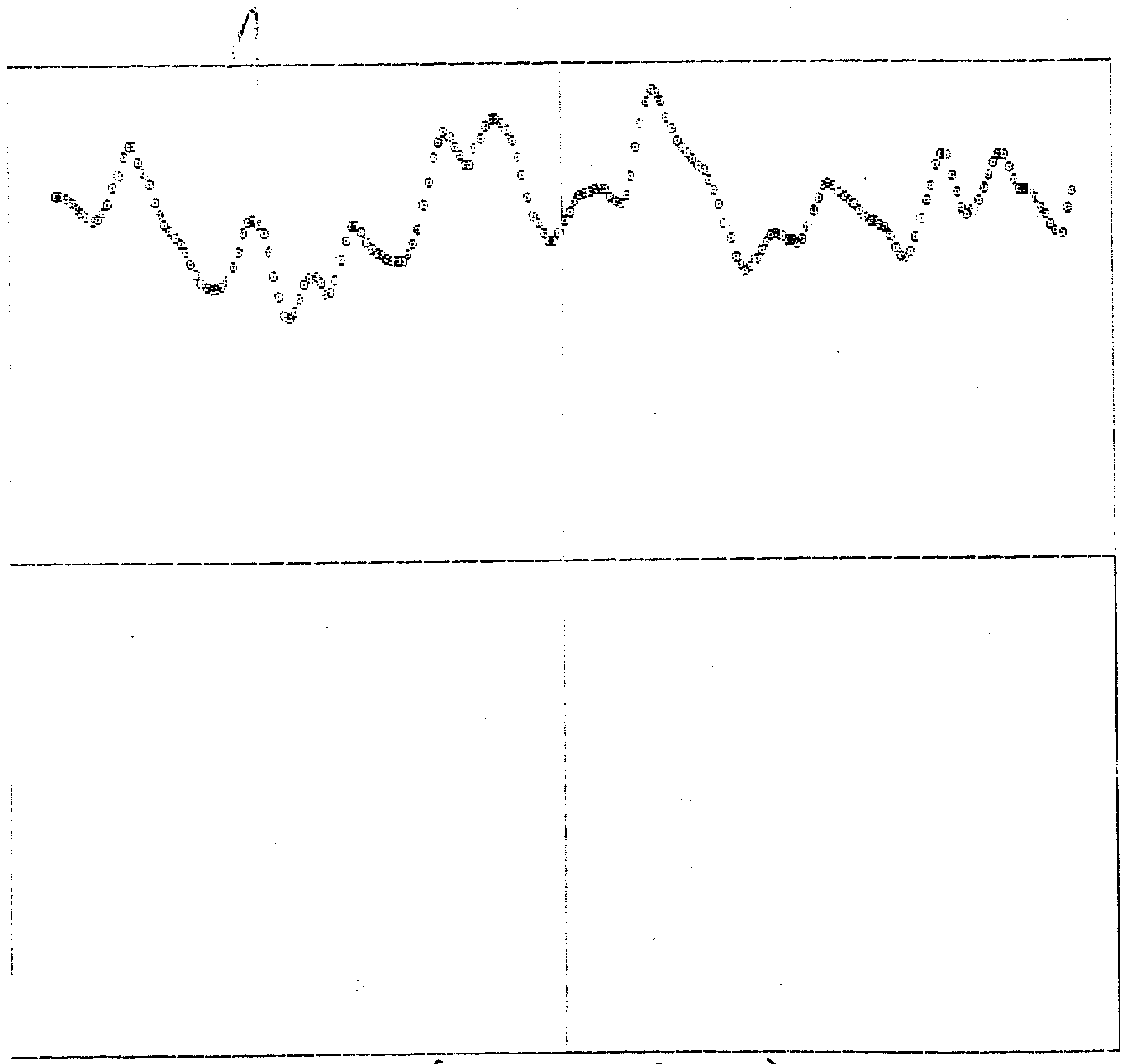


Fig. 9 (Original Graph)

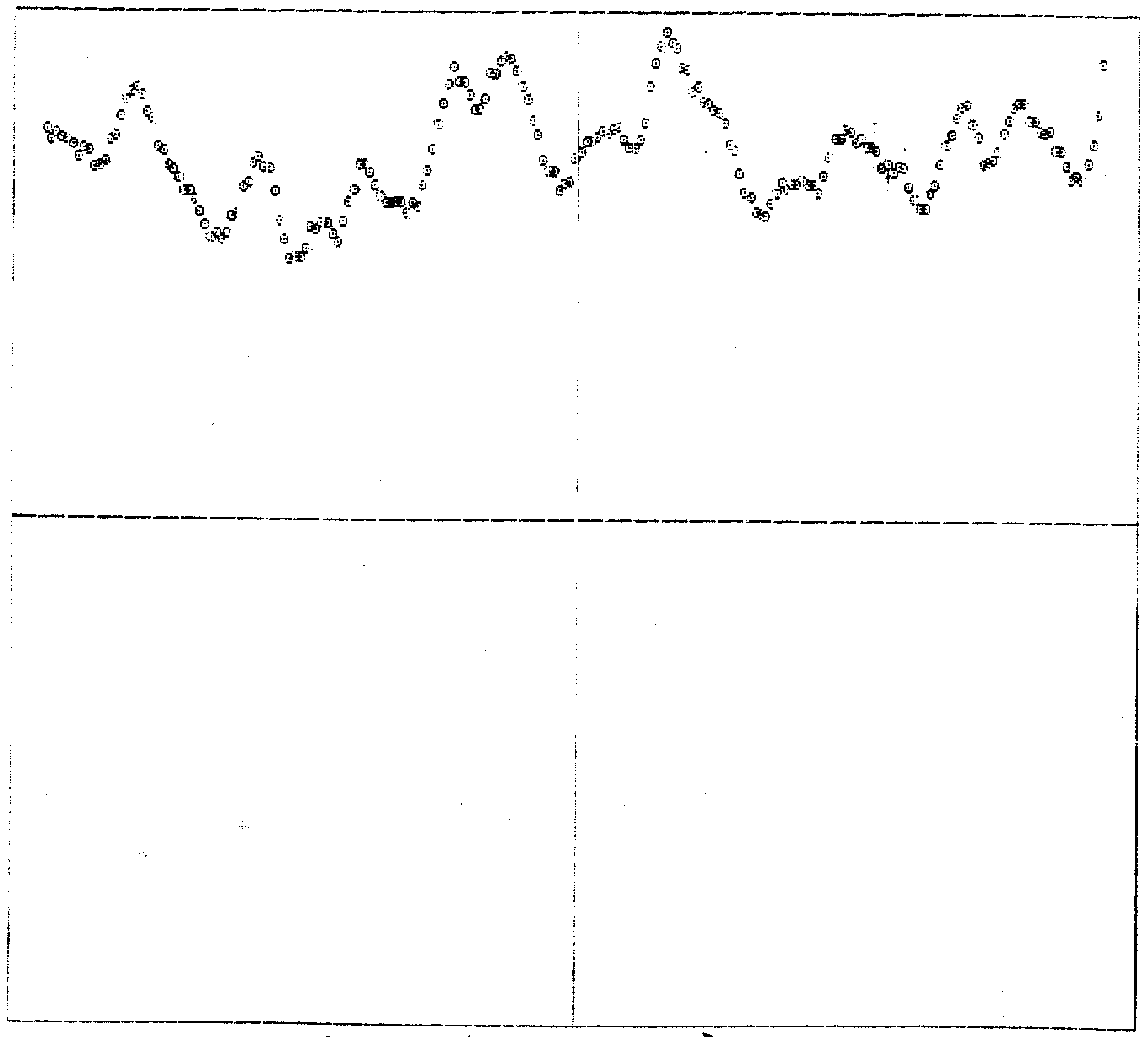


Fig. 10 (AR Process)

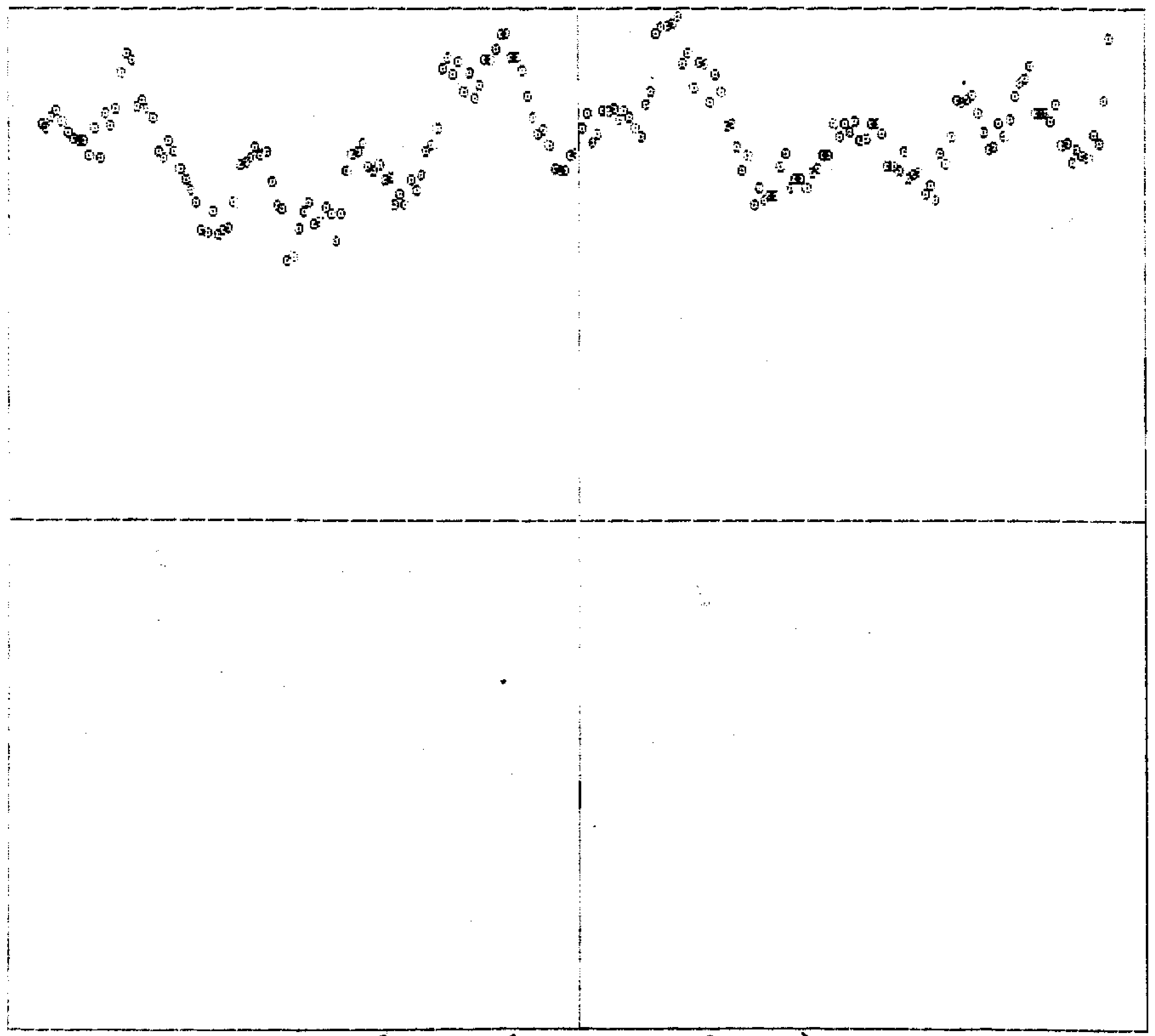


Fig. 11 (Wavelet Network)

9.2 Experiments with Real Data

The real data used in this case is the well known gas furnace data from Series J of [BoxJ'70]. The function to be approximated is the percentage concentration of CO_2 given in that data.The data was taken for 200 points. The plotted graph is shown in Fig. 9.

For the AR process the best results(minimum error) was achieved with a 3^{rd} order process.The approximated function was plotted and the result appears in Fig. 10. The AR coefficients are given below :

$$\phi_1 = 1.970916, \quad \phi_2 = -1.372316, \quad \phi_3 = 0.339978.$$

In the Multiple Mother Wavelet Network the network was trained with 16 wavelets taking 8 each from two different families whose functional forms are given below :

Family 1:

$$g^1_{trans,scale}(x) = -xe^{-\frac{1}{2}x^2}$$

Family 2:

$$g^2_{trans,scale}(x) = sin(x)e^{-\frac{1}{2}x^2}$$

The values of the translation and the scale parameters included those values that were present in the original function.The approximated function was plotted and the result is shown in Fig. 11. The values of the various transform coefficients are given below.

$$w^1_{0.17,10.15} = 4.896723 \quad w^2_{0.24,10.11} = 5.789123$$

$$w^1_{-5.33,5.02} = 5.021989 \quad w^2_{-5.19,5.08} = 6.907865$$

$$w^1_{7.67,2.20} = -6.209817 \quad w^2_{7.23,2.19} = -5.902156$$

$$w^1_{-2.45,2.27} = 5.302910 \quad w^2_{-2.25,2.19} = 4.123410$$

$$w^1_{2.55,2.29} = -8.291540 \quad w^2_{2.69,2.38} = -7.980543$$

$$w^1_{5.49,5.44} = 4.901213 \quad w^2_{5.56,5.31} = -4.091357$$

$$w^1_{7.31,2.53} = 3.481243 \quad w^2_{7.36,2.57} = -8.975621$$

$$w^1_{10.06,1.20} = 9.197865 \quad w^2_{10.37,1.25} = 2.901879$$

In the case of both the Unconstrained GA-based algorithm and the Penalty-based algorithm the *dictionary* included a total of 45 wavelets taking 15 each from 3 different functional families of wavelets. The functional forms for these families are given below :

Family 1:

$$g^1_{trans,scale}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \end{cases}$$

Family 2:

$$g^2_{trans,scale}(x) = -xe^{-\frac{1}{2}x^2}$$

Family 3:

$$g^3_{trans,scale}(x) = sin(x)e^{-\frac{1}{2}x^2}$$

The values of the translation and the scale parameters included those values that were present in the original function. The algorithm was run for 100 generations.

Fig. 12 (Unconstrained GA)

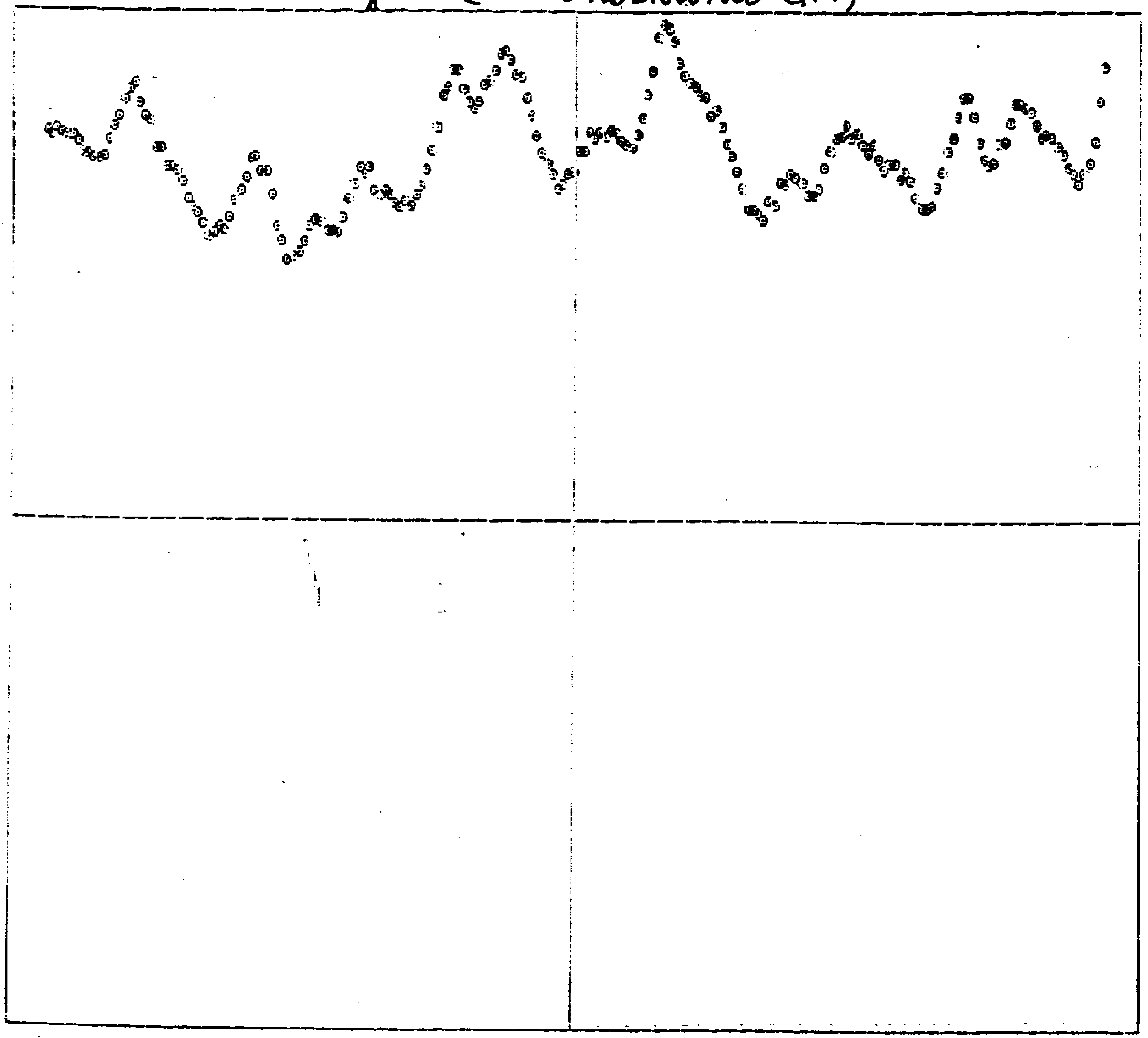
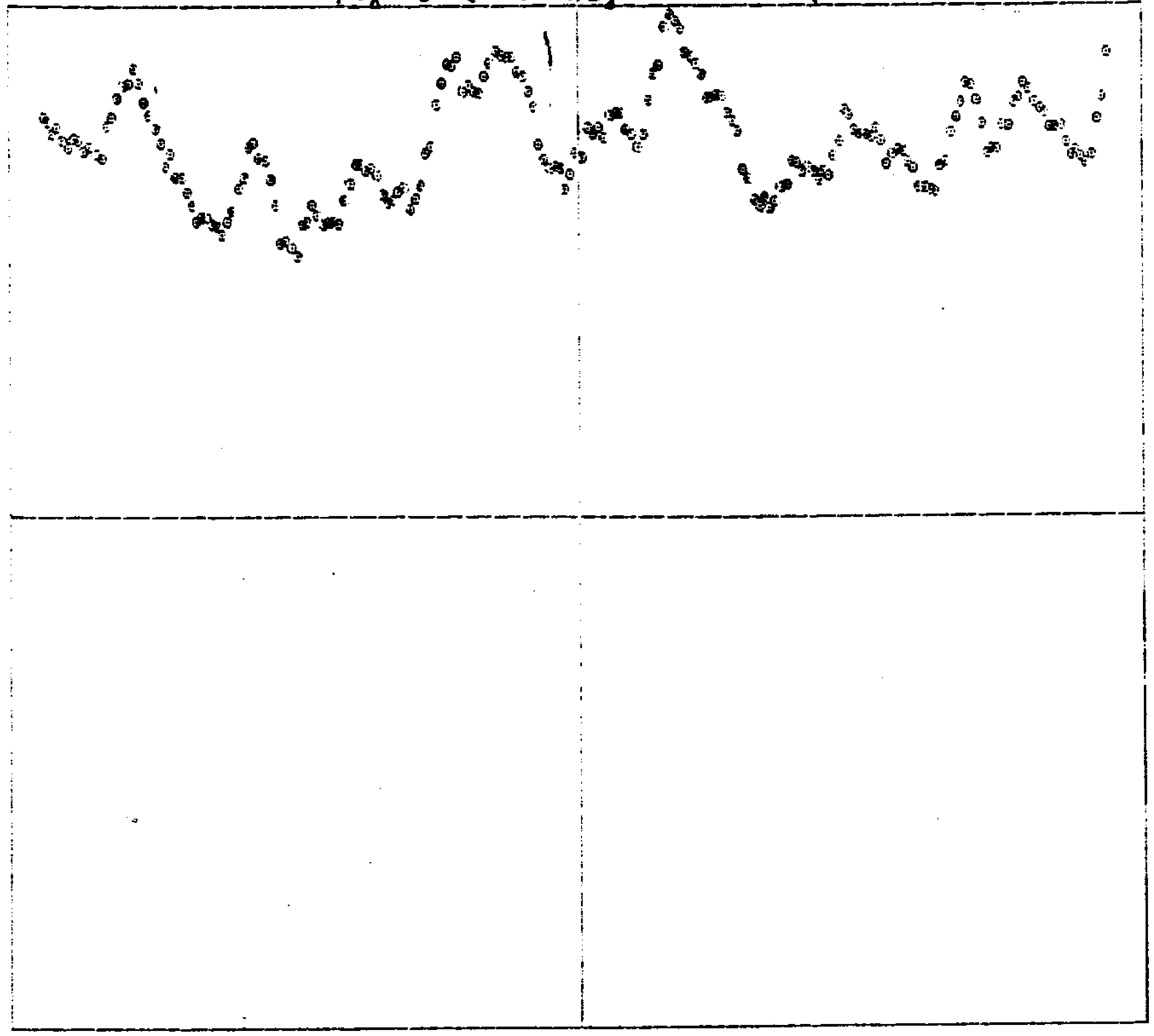


Fig. 13 (Penalty-based GA)



The approximated functions for both the cases were plotted and the results are shown in Figs. 12 and 13. The best fit strings and the maximum, minimum and average fitness values for both the cases (unconstrained and penalty-based) are given below.

For the unconstrained GA method :
Maximum fitness : 27.678931
Minimum fitness : 0.157290
Average fitness : 2.409567
Best fit string : 101011101001011101001011100011111010111010011

For the penalty-based GA method :
Maximum fitness : 10.391239
Minimum fitness : 0.302199
Average fitness : 1.154093
Best fit string : 101001101001001101001001100010111001100100011

Note the marked difference in the number of 1's between the two cases.The fitness values are also different due to the penalty term used. Also comparing Figs. 12 and 13 we find that the curve in Fig. 12 approximates the original curve (Fig. 9) than that in Fig. 13. This is because in the penalty-based model certain wavelets with relatively low values of transform coefficients have been forcefully removed from the best-fit string to maximise the fitness. This may be one drawbacks of this penalty-based model.

10 Conclusion

In this article a known statistical technique for function the (AR method) forecasting is first studied. Then a recently discovered Wavelet Network model is investigated in detail. This Wavelet Network model is improved further by modifying the model to incorporate multiple mother wavelets. The defects of such Wavelet Network models are then investigated and a Genetic Algorithm-based model using wavelet decomposition is proposed and implemented . Finally the defects of this GA-based model are highlighted and the model is modified giving a Penalty-based GA model.

11 References

[Akai'74] H Akaike.
"A new look at the statistical model identification.
IEEE Trans. in Auto. Control, AC-19 , 1974.

[Bell'61] R Bellman.
Adaptive Control Processes : A Guided Tour.
Princeton University Press , Princeton , New Jersey , 1961.

[BoxJ'70] G.E.P Box and G Jenkins.
Time Series Analysis : Forecasting and Control.
Cambridge University Press , Cambridge , 1970.

[Daub'90] I Daubechies.
"The wavelet transform , time-frequency localisation and signal analysis."
IEEE Trans. on Inform. Theory. , vol 36 , Sept. 1990.

[Feyn'65] R.P Feynman.
Path Integrals in Quantum Mechanics.
McGraw Hill , New York , 1965.

[Gold'89] D.E Goldberg.
Genetic Algorithms in Search , Optimisation , and Machine Learning.
Addison-Wesley Publishing Co.,Inc., Reading , 1989.

[Hann'82] E.J Hannan and J Rissanen.
"Recursive Estimation of mixed Autoregressive Moving Average methods."
Biometrika , 69 , 1982.

[HecN'89] R Hecht-Nielsen.
NeuroComputing.
Addison Wesley Publishing Co., Inc., Reading , 1989.

[Pati'93] Y.C Pati and P.S Krishnaprasad.
"Analysis and synthesis of feedforward neural networks using discrete affine wavelet transformations."
IEEE Trans. on Neural Net. , Vol 4 , No. 1 , Pg 73-85 , 1993.

[Schw'78] G Schwarz.
"Estimating the dimension of a model."
Annals of Stat. , 6 , 1978.

[Tere'90] T.C Mills.
Time Series Techniques for Economists.
Cambridge University Press , Cambridge , 1990.

[Zhan'92] Q Zhang and A Bienveniste.
"Wavelet Networks".
IEEE Trans. on Neural Nets. , Vol 3 , No. 6 , Pg Nov 1992.