Open Journal of Statistics, 2021, 11, 673-689
https://www.scirp.org/journal/ojs
ISSN Online: 2161-7198
ISSN Print: 2161-718X

# ANN-Time Varying GARCH Model for Processes with Fixed and Random Periodicity

**Elias K. Karuiru[1], John Mwaniki Kihoro[2], Thomas Mageto[3], Anthony Gichuhi Waititu[3]**

[1]The Pan African University Institute for Basic Sciences, Technology and Innovation (PAUSTI), Nairobi, Kenya
[2]School of Computing and Mathematics, Co-Operative University, Nairobi, Kenya
[3]Department of Statistics and Actuarial Sciences, JKUAT, Nairobi, Kenya
Email: karuirue@gmail.com

**How to cite this paper:** Karuiru, E.K., Kihoro, J.M., Mageto, T. and Waititu, A.G. (2021) ANN-Time Varying GARCH Model for Processes with Fixed and Random Periodicity. *Open Journal of Statistics*, **11**, 673-689.
https://doi.org/10.4236/ojs.2021.115040

**Received:** August 16, 2021
**Accepted:** October 5, 2021
**Published:** October 8, 2021

## Abstract

Financial Time Series Forecasting is an important tool to support both individual and organizational decisions. Periodic phenomena are very popular in econometrics. Many models have been built aiding capture of these periodic trends as a way of enhancing forecasting of future events as well as guiding business and social activities. The nature of real-world systems is characterized by many uncertain fluctuations which makes prediction difficult. In situations when randomness is mixed with periodicity, prediction is even much harder. We therefore constructed an ANN Time Varying Garch model with both linear and non-linear attributes and specific for processes with fixed and random periodicity. To eliminate the need for time series linear component filtering, we incorporated the use of Artificial Neural Networks (ANN) and constructed Time Varying GARCH model on its disturbances. We developed the estimation procedure of the ANN time varying GARCH model parameters using non parametric techniques.

## Keywords

Fixed Periodicity, Random Periodicity, Artificial Neural Network, Time Varying GARCH

## 1. Introduction

Periodic phenomena such as business and temperature cycles appear in our daily life very often. Many models have been built aiding capture of these periodic trends as a way of enhancing forecasting of future events as well as guiding business and social activities. The nature of real world systems is characterized by many uncertain fluctuations which makes prediction difficult. Prediction is made much harder when there is mixed periodicity and randomness especially when noise is very large and periodicity is covered up. Currently, time series analysis involves describing the observed values as a path of a stationary process by fitting the data using different non-structural models. One of the most used models is the Autoregressive Moving Average (ARMA) model. ARMA model was introduced by Whittle in 1951 and was popularised by BOX and Jenkins in 1970. The model describes a time series as a stationary stochastic process which linearly combines an autoregressive polynomial of the current data to past data and a moving average polynomial of history noises up to lag q. When the time series is not stationary then de-trending and de-seasoning is needed. Considering the classical time series decomposition

$$Y(t)=T(t)+S(t)+C(t)+X(t) \tag{1}$$

where $Y(t)$ is the observed data, $T(t)$ is the trend component, $S(t)$ is the seasonal component, $C(t)$ is the cyclic component and $X(t)$ is the noise term. The ARMA model is applied to fit the noise term [1].

This method is quite limited due to inflexible assumption placed on the seasonal component. Franses in [2] recommends that much attention should be placed in the seasonal fluctuations when dealing with econometric data. This is caused by randomness in the seasonal variation that can more precisely explain the behaviour of economic agents. Franses in [2] also observed that the non-seasonal fluctuations in many periodic observed macroeconomic time series are non stationary over time and hence insinuating possibility of dependence between seasonal and non seasonal fluctuations in some time series. The classical decomposition model is hence seen to be limited. In-order to solve this problem, many sophisticated models have been built based on the ARMA model.

Autoregressive Conditional Heteroskedasticity (ARCH) model is also another commonly used model in financial time series. ARCH is a stationary non linear model. An ARCH(q) first models the stationary process by an AR(q) model, and takes the variance of the residual term as a q-th autoregressive polynomial relating to the history squared residuals back to lag q. The model was first proposed by Engle in 1982 and latter generalised by Bollerslev and Taylor in 1986 and termed as Generalised Autoregressive Conditional Heteroskedasticity (GARCH) model.

These models have improved the application range to the real world problems in time series. However, in periodic phenomena, randomness and periodicity estimation is still considered separately. The GARCH model for example fits a periodic function to the seasonal trend and adds it to the ARMA type process of the residuals linearly. This is hence seen as a limitation of the model.

The concept of random periodic processes describes the randomness and periodicity in stochastic process revolution simultaneously. It therefore inspires us to apply random periodic process to classical time series analysis as a way of describing the random periodic phenomena.

## 2. Statement of the Problem

Financial Time Series Forecasting is an important tool to support both individual and organizational decisions. The existing forecasting models which include the ARMA and ARCH models and their various forms are built under the linearity and non linearity assumption respectively. This assumption creates a limitation in forecasting complex real world problems which exhibit both linear and non linear properties. Further, these existing models concentrate only on random processes without considering that some processes exhibit random periodic properties. In addition, some processes also exhibit a mixture of fixed and random periodicity. This study therefore constructed a model with both linear and non-linear attributes and specific for processes with fixed and random periodicity. To eliminate the need for time series linear component filters, we incorporated the use of Artificial Neural Networks (ANN) which requires less background knowledge about the data.

ANN-Time Varying GARCH model for processes with fixed and random periodicity will be applied in financial markets whose products fluctuate periodically and also affected by other periodic but random factors. For retailers, for instance, the model may reveal consumer demand for winter clothes spikes at a distinct time period each year, information that would be important in forecasting production and delivery requirements.

## 3. Literature Review

Financial time series analysis involves development of a model that describes the relationship between the variable past observations. This model is then used to predict the future when the time series is extrapolated. This approach is applied when limited knowledge on the process of generating data is available or there is no satisfactory explanatory model that relates the prediction variable to the explanatory variables [3]. Time series forecasting have attracted many researchers leading to development and improvement of many models. In time series forecasting, model selection is very critical since it determines the accuracy of the forecasts. The determination of whether to fit a linear or non linear model is dependent on the relationship between the current value and the past observations [1].

The concept of random periodicity was introduced by Feng and Zhao in the year 2015. They discovered that many periodic real world systems are subject to the influence of randomness. They proposed that the mix between periodicity and randomness may be best described by random periodic motion rather than a periodic motion. Liu in [4] regarded stationary process as a special kind of random periodic processes. Random periodic processes describe a mixed structure of seasonal and random patterns [5]. Liu in [4] used random periodic process to replace stationary process in the analysis of time series as a way of enlarging the application and enhance accuracy of ARMA model estimation. Zhao in [6] established an ergodic theory in the random periodic regime with periodic measures and random periodic processes.

In his dissertation, [4] gave a counter example situation in which the classical times series additive structure is not sufficient as there is seasonal pattern in the irregular component. In the example, the de-seasoned data was stationary and satisfied an autoregressive equation. The autocorrelations and partial autocorrelations of residuals were near zero, which indicated the independency of noise. However, the autocorrelations and partial autocorrelations of squared residuals showed obvious periodic pattern, which indicated sufficient dependency of time in the volatility of noise. He also observed that the Shapiro-Wilk test rejected the "normal" hypothesis with very small p-value. But if tested on the periodic-point sequence of residuals, he observed that the Shapiro-Wilk test accepted that it satisfied a normal distribution. He therefore concluded that the standard procedure to analyze a time series by the classical decomposition fails to obtain the correct result for a mixed season and randomness case. He hence developed an ARMA model for random periodic process to model the seasonal and irregular component simultaneously.

We extend the work of [4] by considering that some processes also exhibit a mixture of fixed and random periodicity. We therefore constructed a model for processes with fixed and random periodicity to model financial time series which exhibit heteroscedastic property. To eliminate the need for time series linear component filters, we incorporated the use of Artificial Neural Networks (ANN) which do requires less background knowledge about the data.

## 4. Methods

**1) Random Periodic Process**

A random periodic process of period $\tau$ of the random dynamic system $\Phi:\mathbb{R}^+\times\Omega\times M\to M$ is an $\mathcal{F}$-measurable map $R:\mathbb{R}^+\times\Omega\to M$ such that for almost all $\omega\in\Omega$,

$$\Phi\left(t,\theta(s)\omega\right)R(s,\omega)=R(t+s,\omega),R(s+\tau,\omega)=R\left(s,\theta(\tau)\omega\right) \tag{2}$$

for any $s,t\in\mathbb{R}$.

For a statistical description, we usually do not know the exact expression of the dynamical system driving the time series. We only consider the second equation in 2 as the definition of random periodic process, while the first part is hidden in the time series evolution [4]. For detailed definition on random periodic path, its properties and periodic measure refer to [5].

**2) Decomposition of Time Series with fixed and Random periodicity**

A time series is normally decomposed into four components namely trend, seasonal, cyclical and random. When these four components are believed to be independent then an additive model is employed. When the components are not necessarily independent then a multiplicative model is assumed [1]. Consider the classical decomposition model of time series

$$Y(t)=T(t)+S(t)+C(t)+X(t) \tag{3}$$

where $Y(t)$ is the observed data, $T(t)$ is the trend, $S(t)$ is the seasonal component $C(t)$ is the cyclical component and $X(t)$ is the random component. In some situations, there is seasonal pattern in the irregular component, in which the classical additive structure is not sufficient [4]. The insufficiency of the classical decomposition to figure out the mixed structure of the seasonal and noise components inspires us to make modification to it. We attempt to use random periodic process to describe the seasonal and noise components simultaneously. The modified model is as follows:

$$Y(t)=T(t)+S_f(t)+C(t)+R(t) \tag{4}$$

where

$$R(t)=S(t)X(t) \tag{5}$$

and $S_f(t)$ is the fixed periodic component. Therefore $R(t)$ is a random periodic process with the following properties:

1) $E\left[R(t)^2\right]<\infty$.

2) $E\left[R(t)\right]$ is a deterministic periodic function of time *t*.

3) The autocovariance function of random periodic process is periodic function of time t on both indexes.

$Y(t)$ is therefore a process with fixed and random periodicity.

## 4.1. Periodic Time Series Models

### Introduction

The introduction of periodic models into economics dates back to the late 1980s. By then, the focus was on describing trending consumption and income data, and the use of periodic models for out-of-sample forecasting.

The fixed periodic models assume that the coefficients of the underlying model are purely repetitive, that is, they vary with the fixed period *s*. That is, the number of the coefficients is *s* times more than the standard coefficients in the standard stochastic model. A fixed periodic model therefore permits a different standard stochastic model for each fixed period *s*. On the other hand, random periodic models have all the coefficients vary with time *t*. That is, all the coefficients are different from each other even if they are in the same random period $\tau$. Therefore, the major difference between fixed and random periodicity is that the same time points in different periods will have the same behaviour in fixed periodicity but there will be small fluctuations between different periods in random periodicity.

The following are the graphical representation of fixed periodicity, random periodicity and a combination of fixed and random periodicity (**Figures 1-3**).

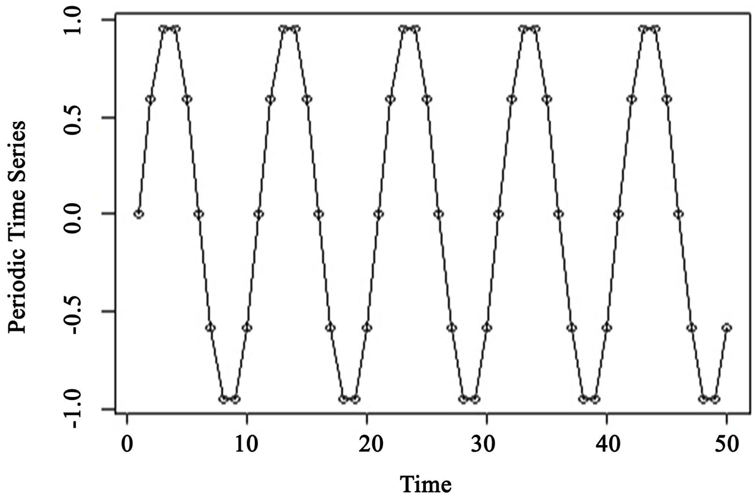

**Figure 1**. Periodic time series with fixed periodicity $s = 10$.

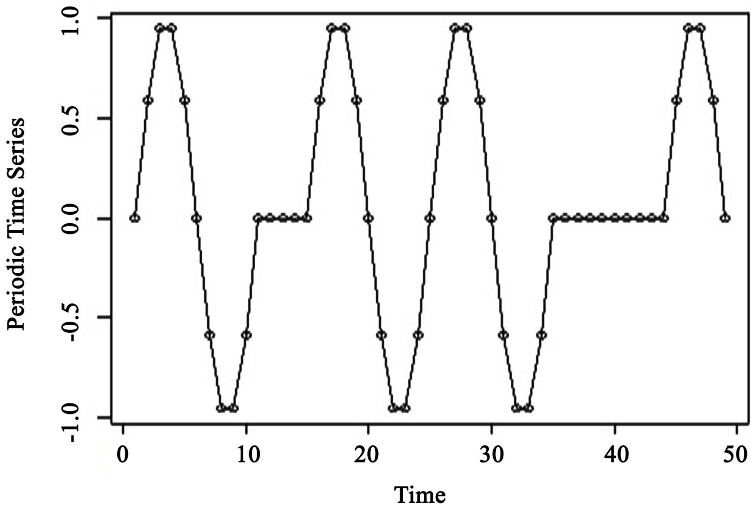

**Figure 2**. Periodic time series with random periodicity $\tau = 10$.

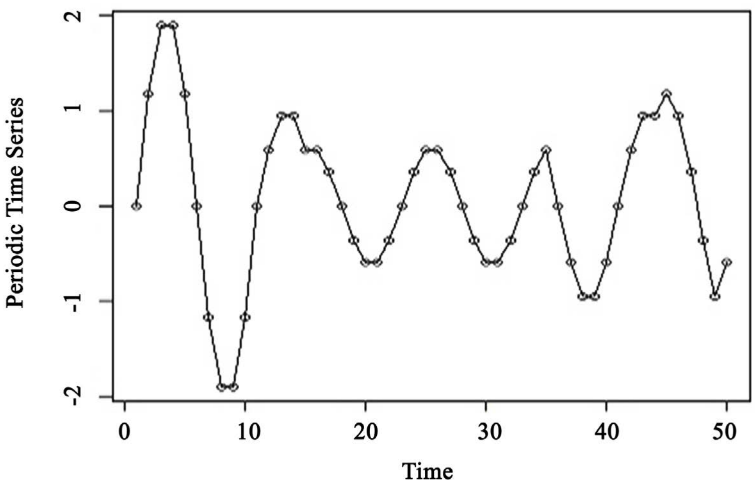

**Figure 3**. Periodic time series with fixed periodicity $s = 10$ and random periodicity $\tau = 10$.

## 4.2. Model for Processes with Fixed and Random Periodicity

### 4.2.1. ANN-Time Varying GARCH Model

Next we construct an ANN-Time Varying GARCH model for processes with fixed and random periodicity. Define the process with fixed periodicity *s* and random periodicity $\tau$ as $Y(t)$ to be an ANN-Time Varying GARCH process, if for each *t*,

$$Y_t = \mu_t + R_t, \tag{6}$$

where

$$\mu_t = \alpha_0 + \sum_{j=1}^{m} \alpha_j f\left\{V_j + \sum_{i=1}^{d} V_{ij} Y_{t-is}\right\}, \tag{7}$$

$$R_t = \sigma_t Z_t, \tag{8}$$

such that

$$\sigma_t = \sqrt{w(t) + \sum_{i=1}^{p} \phi_i(t) R_{t-i}^2 + \sum_{i=1}^{q} \theta_i(t) \sigma_{t-i}^2} \tag{9}$$

and $Z_t$ is an i.i.d white noise. Therefore, the model is an ANN model with Time Varying GARCH disturbances. Where,

1) $\alpha_j$ ( $j = 1, 2, \cdots, m$ ) and $V_{ij}$ ( $j = 1, 2, \cdots, m$ and $i = 1, 2, \cdots, d$ ) are the hidden and input connection weights respectively.

2) $\alpha_0$ and $V_j$ ( $j = 1, 2, \cdots, m$ ) are the output and hidden layers connection bias respectively.

3) $d$ is the number of nodes in the input layer.

4) $m$ is the number of nodes in the hidden layer.

5) $f$ is the hidden layer transfer function.

6) $R_t$ is a random periodic process.

7) $w(.)$, $\phi(.)$ and $\theta(.)$ are non negative functions of time *t*.

Hence, the ANN model of (7) in fact performs a nonlinear functional mapping from the past observations $Y_{t-s}, Y_{t-2s}, Y_{t-3s}, \cdots$ to the future value $Y_t$ therefore making connection weights of the model vary with period *s*.

Equation (8) is a Time Varying GARCH(p, q) process with time varying parameters $w(t), \phi_i(t)$ and $\theta_i(t)$.

The estimation of parameters for the model will be done using non parametric techniques.

### 4.2.2. Assumptions of the Model

1) $Y(t)$ is a non-stationary process which can be decomposed as follows:

$$Y(t) = T(t) + S(t) + C(t) + R(t) \tag{10}$$

where $T(t)$ is the trend, $S_f(t)$ is the fixed periodic component with period *s*, $C(t)$ is the cyclical component and $R(t)$ is the random periodic component with period $\tau$.

2) $R(t)$ is a random periodic process with the following properties:

a) $E\left[R(t) \mid \mathcal{F}_{t-1}\right] = 0$.

b) $Var\left[R(t)\mid\mathcal{F}_{t-1}\right]=\sigma_t^2$.

c) The autocovariance function of random periodic process $R(t)$ is periodic function of time t on both indexes.

$$\gamma_R(t,r)=\gamma_R(t+\tau,r+\tau) \tag{11}$$

for any $t,r\in\mathbb{Z}$

Proof:

$$\begin{aligned}\gamma_R(t+\tau,r+\tau)&=\mathbb{E}\Big[\big(R(t+\tau,\omega)-m(t+\tau)\big)\big(R(r+\tau,\omega)-m(r+\tau)\big)\Big]\\&=\mathbb{E}\Big[\big(R(t,\theta(\tau)\omega)-m(t)\big)\big(R(r,\theta(\tau)\omega)-m(r)\big)\Big]\\&=\mathbb{E}\Big[\big(R(t,\omega)-m(t)\big)\big(R(r,\omega)-m(r)\big)\Big]\\&=\gamma_R(t,r).\end{aligned} \tag{12}$$

3) The conditional variance $\sigma_t^2$ of $R_t$ given information available up to time $t-1$, has an autoregressive structure and is positively correlated to its own recent past and to recent values of the squared returns $R_t^2$. This captures the idea of volatility (conditional variance) being "persistent": large (small) values of $R_t^2$ are likely to be followed by large (small) values.

## 4.3. Determination of Parameters

### 4.3.1. Determination of ANN Parameters

The Artificial Neural Network model has a number of parameters to be estimated prior to its application. The topology of the ANN is as follows Figure 4.

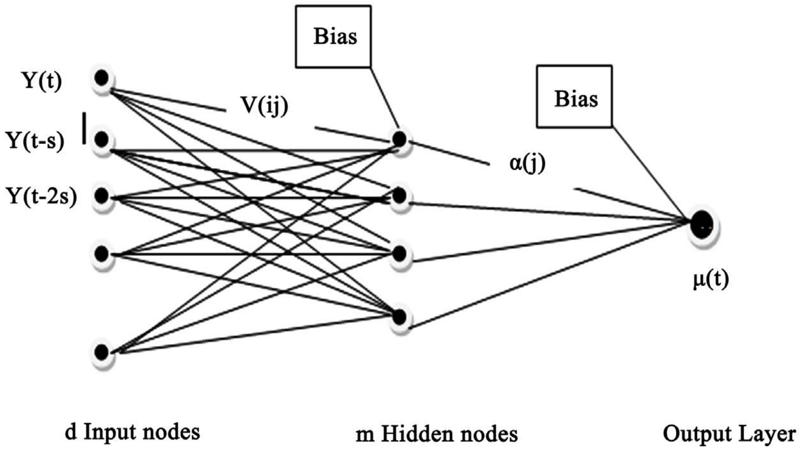

Figure 4. Topology of artificial neural network.

Further, an appropriate activation function needs first to be selected before the parameters are estimated. These parameters include biases, weights connecting the inputs with the hidden nodes $V_{ij}$ ($j=1,2,\cdots,m$ and $i=1,2,\cdots,d$), weights linking the hidden nodes to the output node $\alpha_j$ ($j=1,2,\cdots,m$), numbers of input nodes $d$ and number of hidden nodes $m$.

Determination and selection of input nodes in the input layer and the number of hidden nodes in the output layer forms the major problems in artificial neural network specification. Problems, which can occur due to poor selection of the parameters, include: increased input dimensionality, increased computational complexity, increased memory requirements, increased learning difficult, mis-convergence, poor model accuracy and problem of understanding results from complex models [7].

**Activation function**

The activation function is a mathematical "gate" in between the input feeding the current neuron and its output going to the next layer [8] [9]. It can be as simple as a step function that turns the neuron output on and off, depending on a rule or threshold. Or it can be a transformation that maps the input signals into output signals that are needed for the neural network to function. The Common Nonlinear Activation Functions include: The sigmoid/logistic function which is defined as:

$$f(x) = \frac{1}{1+\exp(-x)} \tag{13}$$

The logistic function has smooth gradient, bounded outputs and clear predictions advantages. However, the function is computationally expensive, not zero centred and has a vanishing gradient. The hyperbolic tangent function which is defined as

$$\tanh(x) = 2\left(\frac{1}{1+\exp(-2x)}\right)-1, \tag{14}$$

unlike the the sigmoid function it is zero centred. Other activation functions include Rectified Linear Unit (ReLu) which is computationally efficient despite having the dying ReLu problem. The Leaky ReLU function despite solving the ReLu dying problem does not provide consistent predictions for negative input values. The parametric ReLu unlike leaky ReLU, this function provides the slope of the negative part of the function as an argument. It is, therefore, possible to perform backpropagation. We however adopt and apply the hyperbolic tangent activation function 14 since it zero centred and bounded.

**Bias and Connection Weights**

When the activation function is defined we select the training/learning algorithm which is appropriate. The parameter space includes the vector of the matrix $W_1 = \left(\left(V_{ij}\right)\right)$ of weights connecting the inputs with the hidden neurons and the matrix $W_2 = \left(\left(\alpha_j\right)\right)$ of weights linking the hidden neurons to the output neuron and a vector of biases *b*. To estimate $\Psi = \left(W_1, W_2, b\right)$, we minimize the following error function.

$$E(\Psi) = \sum_{i=1}^{N}\left(Y_t - \hat{Y}_t\right)^2 \tag{15}$$

The optimization techniques for minimizing the error function 15 are referred to as learning rules [7]. The generalised delta rule method also referred to as the error back propagation is the best known method which continuously modifies connection weight to reduce the difference between the required and the actual output. For the learning process, the data must be divided in two sets: the training data set, which is used to calculate the error gradients and to update the weights; and the validation data set, which allows selecting the optimum number of iterations to avoid overlearning.

In this method the weights are adjusted as follows:

$$\begin{aligned} V^{r+1} &= V^{r} + \Delta V \\ \alpha^{r+1} &= \alpha^{r} + \Delta \alpha \end{aligned} \tag{16}$$

We have the $r^{\text{th}}$ iteration weights as:

$$\alpha_j^{r+1} = \alpha_j^{r} - \lambda_1 \frac{\delta E\left(\Psi\right)}{\delta \alpha_j}, \tag{17}$$

for $j = 1, \cdots, m$ and $\lambda_1$ is the step gain.

Similarly,

$$V_{ij}^{r+1} = V_{ij}^{r} - \lambda_2 \frac{\delta E\left(\Psi\right)}{\delta V_{ij}}, \tag{18}$$

for $i = 1, \cdots, d; j = 1, \cdots, m$ and $\lambda_2$ is the step gain. The weights are adjusted until a stopping criterion is met. This method is however slow and unstable.

This rule has various variations such as gradient descent with momentum, gradient descent with adaptive learning rate, quasi-Newton, conjugate gradient, Scaled conjugate gradient and Levenberg-Marquardt. In spite of various modified back propagation training algorithms, some crucial limitations of the standard back propagation technique such as slow convergence rate still remain unresolved [3]. Standard batch backpropagation is the most popular training method of all, but it is slow, unreliable, and requires the tuning of the learning rate, which can be a tedious process [7]. We hence discuss and apply the Quasi Newton method which is very fast and reliable.

The Quasi Newton method starts by inputing an initial set of weights $\Psi^0$ from which $S^2\left(Y_i, \Psi^0\right)$ is determined. From the second order Taylor operation principles $S^2\left(Y_1, \Psi^1\right)$ can be found.

$$\Psi^1 = \Psi^0 - B_0^{-1} A_0 \tag{19}$$

where $B_0^{-1} A_0$ represents the change direction. $B_0$ represents the direction angle while $A_0$ represents the direction size. The direct off-diagonal elements of the matrix $B_0$ are evaluated as

$$B_0 = \frac{\delta^2 S^{2\left(Y_i, \Phi^0\right)}}{\delta \Phi^{0,d} \delta \Phi^{0,m}}. \tag{20}$$

While the direct diagonal elements of matrix $B_0$ are evaluated as

$$B_0 = \frac{\delta^2 S^{2\left(Y_i, \Phi^0\right)}}{\left(\delta \Phi^{0,d}\right)^2}. \tag{21}$$

The minimization then continuous from iteration 1 to 2 until the stopping criterion is met. Generally, the iterations are given as:

$$\Psi^{r+1} = \Psi^{r} - B_r^{-1} A_r . \tag{22}$$

However, the Hessian matrix $B$ may become non-singular. This means that $B_r^{-1}$ would be undefined. The method solves the problem by numerically approximating $B_r$. The method redefines the general equation as

$$\Psi^{r+1} = \Psi^{r} - M_r B_r^{-1} A_r , \tag{23}$$

where $M_r$ is the step length. By letting,

$$a_r = \Psi^{r+1} - \Psi^{r} = M_r B_r^{-1} A_r , \tag{24}$$

represent the change in parameters in the $r^{\text{th}}$ iteration and

$$b_r = A^{r+1} + A^{r} . \tag{25}$$

The method has

$$B_{r+a} a_r = b_r . \tag{26}$$

Therefore, $B_{r+1}$ is the ratio of the change in the gradient to the change in the parameters.

The Stopping rules commonly used are:

1) $\left|\Phi^{r+1} - \Phi^{r}\right| < \varepsilon$, for $\varepsilon > 0$ but small.

2) $\left|S\left(Y, \Phi^{r+1}\right) - S\left(Y, \Phi^{r}\right)\right| < \varepsilon$, for $\varepsilon > 0$ but small.

3) $S\left(Y, \Phi^{r+1}\right)$ is less than a specified lower bound.

4) $r$ is greater than a specified number of iterations.

It is worth noting that rule (4) can be applied together with rule (1), rule (2) and rule (3).

**Number of nodes in Input Layer**

The number of input nodes is the number of lags that will be considered as inputs in the neural network input layer. To determine which lags to include in an input set Y of variables, autocorrelations and partial autocorrelations analysis together with AIC and its variation have been used, but they have not been very helpful [7]. Other methods such as network pruning have been developed but they have a problem associated with computation of the elements of the Hessian matrix which are the second derivatives of the error function with respect to the training weights.

The automatic relevance determination method developed by Mackay (1992) and Neal (1996) defines a prior over the regression parameters that embody the concept of uncertain relevance, so that the model is effectively able to infer which variables are relevant and then switch the others off thus preventing those inputs from causing significant over-fitting [7]. This method may be applied to any univariate time series problem if an autoregressive neural network (ARNN) is to be fitted to the data. An improvement of Mackay's ARD method uses a sub set of the weights, with a clear cut method of selecting the set to ensure that the effects of a particular input lag are well represented [10]. This method is computationally fast, as it requires no evaluation of eigenvalues of the Hessian matrix, which is not easy [7].

### Number of nodes in Hidden Layer

A multi layer feedforward network with at least one hidden layer and a sufficient number of hidden nodes is powerful enough to represent any form of time series [7]. Deciding the number of neurons in the hidden layers is a very important part of deciding your overall neural network architecture. Using too few neurons in the hidden layers will result in something called underfitting. Underfitting occurs when there are too few neurons in the hidden layers to adequately detect the signals in a complicated data set. Using too many neurons in the hidden layers can result in several problems. First they may result in overfitting. Overfitting occurs when the neural network has so much information processing capacity that the limited amount of information contained in the training set is not enough to train all of the neurons in the hidden layers [11]. Secondly, large number of neurons in the hidden layers can increase the time it takes to train the network.

Several researchers have proposed some rules of thumb for determining an optimal number of hidden units for any application. One rule of thumb is for the size of this hidden layer to be somewhere between the input layer size and the output layer size [12]. Blum in [12] also suggested that the size of the hidden layer should never be more than twice as large as the input layer. Another rule of thumb is that there should be as many hidden nodes as dimensions needed to capture 70 - 90 percent of the variance of the input data set [6]. However, most of those rules are not applicable to most circumstances as they do not consider the training set size and the complexity of the data set to be learnt [11].

Another selection algorithm, a critical value is chosen arbitrarily first. The final structure is built up through the iteration that a new node is created in the hidden layer when the training error is below the critical value [13]. On the other hand, [14] proposed an approach which is similar to [13] but removes nodes when small error values are reached. Rivals in [15] also provided a selection procedure for neural networks based on least squares estimation and statistical tests. Xu and Chen, in [11] developed a novel mathematical method for determining the number of hidden nodes in the hidden layer.

$$m = C\left(N/\left(d\log N\right)\right)^{1/2} \tag{27}$$

where $m$ is the number of hidden nodes, d is the input dimension of the target function, $N$ is the number of training pairs, and $C$ is a constant which does not depend on any function. Another method is the geometric pyramid rule which roughly approximates the number of hidden neurons using the following function:

$$m = \sqrt{d * O}, \tag{28}$$

where $m$ is the number of hidden neurons, $d$ is the number of input nodes and $O$ is the number of outputs.

The most commonly used rule in determining the size of hidden layer is to train a network successfully with one hidden node then two and so on as one monitors the error for the validation data set. This error decreases with every increment in number of hidden nodes $m$ until overfitting begins. At this point training is stopped and this $m$ is taken to be the best choice.

#### 4.3.2. Time Varying GARCH Parameters

Next we consider the following model

$$R_t = \sigma_t Z_t, \tag{29}$$

such that

$$\sigma_t = \sqrt{w(t) + \sum_{i=1}^{p} \phi_i(t) R_{t-i}^2 + \sum_{i=1}^{q} \theta_i(t) \sigma_{t-i}^2} \tag{30}$$

and $Z_t$ is an i.i.d white noise.

Equation (29) is a time varying GARCH(p, q) model. As a process of obtaining meaningful asymptotic theory, the domain of parameter functions of Equation (29) is rescaled to unit interval. Therefore, we study the following process,

$$\sigma_t = \sqrt{w\left(\frac{t}{n}\right) + \sum_{i=1}^{p} \phi_i\left(\frac{t}{n}\right) R_{t-i}^2 + \sum_{i=1}^{q} \theta_i\left(\frac{t}{n}\right) \sigma_{t-i}^2}, \quad t = 1, 2, \cdots, n \tag{31}$$

The stochastic process $R_t$ is therefore said to be a Time Varying GARCH if it satisfies 31. The functions $w(u), \phi(u), \theta(u) \geq 0 \quad \forall u \in (0,1]$ ensures non-negativity of $\sigma_t^2$. We define $w(u), \phi(u), \theta(u) = 0$ for $u < 0$. Rescaling techniques as such does not affect estimation procedure and are common in non parametric regression [10].

Rohan and Ramanathan in [16] developed a two step local polynomial non parametric estimation procedure for time varying GARCH(1, 1) process. We therefore generalise the local polynomial non parametric procedure to estimate the parameters for time varying GARCH(p, q) process.

We first state the following technical assumptions:

1) There exists $\delta > 0$ such that

$$0 < \sum_{i=1}^{p} \phi_i(u) + \sum_{i=1}^{q} \theta_i(u) \leq 1 - \delta, \ \forall\, 0 < u \leq 1 \tag{32}$$

2) There exist finite constants $M_1, M_2, \cdots, M_{1+p+q}$ such that $\forall u_1, u_2 \in (0,1]$,

$$\left| w(u_1) - w(u_2) \right| \leq M_1 \left| u_1 - u_2 \right|$$

$$\left| \phi_1(u_1) - \phi_1(u_2) \right| \leq M_2 \left| u_1 - u_2 \right|$$

$$\vdots$$

$$\left| \phi_p(u_1) - \phi_p(u_2) \right| \leq M_{1+p} \left| u_1 - u_2 \right|$$

$$\left| \theta_1(u_1) - \theta_1(u_2) \right| \leq M_{2+p} \left| u_1 - u_2 \right|$$

$$\vdots$$

$$\left| \theta_q(u_1) - \theta_q(u_2) \right| \leq M_{1+p+q} \left| u_1 - u_2 \right|$$

Assumption (i) is similar to stationarity condition for GARCH($p$, $q$) model. A well defined and unique solution for time varying GARCH($p$, $q$) process requires this condition to hold. The local stationarity of the time varying GARCH(*p*, *q*) requires the Lipschitz continuity condition for the parameters defined in assumption (ii). We do not make any assumptions on the density function of $R_t$.

We define a stationary GARCH(*p*, *q*) process which can locally approximate the original time varying GARCH(*p*, *q*) process 31 in the neighbourhood of a fixed point $U_0$. Let $R_t(u_0), u \in (0,1]$ be a process with the following properties;

1) $E\left[R_t(u_0) \mid \mathcal{F}_{t-1}\right] = 0$.

2) $Var\left[R_t(u_0) \mid \mathcal{F}_{t-1}\right] = \sigma_t^2(u_0)$.

Then, $R_t(u_0)$ is said to follow a stationary GARCH(*p*, *q*) process which is associated with 31 at time point $u_0$ if it satisfies;

$$R_t(u_0) = \sigma_t(u_0) Z_t, \tag{33}$$

such that,

$$\sigma_t(u_0) = \sqrt{w(u_0) + \sum_{i=1}^{p} \phi_i(u_0) R_{t-i}^2 + \sum_{i=1}^{q} \theta_i(u_0) \sigma_{t-i}^2}, \quad t = 1, 2, \cdots, n \tag{34}$$

which is a stationary ergodic process if assumption (1) holds. Using recursive substitution and strong law of large numbers it can be shown that the process 34 has a well defined solution given by

$$\hat{\sigma}_t^2(u_0) = w(u_0) + \sum_{a=1}^{\infty} \prod_{b=1}^{a} \left( \sum_{i=1}^{p} \phi_i(u_0) Z_{t-b}^2 + \sum_{j=1}^{q} \theta_j(u_0) \right) w(u_0), \tag{35}$$

such that $\left|\sigma_t^2 - \hat{\sigma}_t^2\right| \to 0$ almost sure, if $\sigma_0^2$ is finite with probability one.

The local polynomial estimation of the time varying GARCH(p, q) can be done in two steps. The first step involves obtaining a preliminary estimate of $\sigma_t^2$ using a time varying ARCH(p) model. The second step involves reaching the estimators of time varying GARCH(p, q) parameter functions.

**Step 1.** We estimate the prior $\sigma_t^2$ using time varying ARCH(p) model;

$$\sigma_t^2 = \sum_{i=1}^{p} \phi_i\left(\frac{t}{n}\right) R_{t-i}^2, \tag{36}$$

which can also be defined as

$$R_t^2 = \phi_0\left(\frac{t}{n}\right) + \sum_{i=1}^{p} \phi_i\left(\frac{t}{n}\right) R_{t-i}^2 + \sigma_t^2\left(Z_t^2 - 1\right). \tag{37}$$

Treating $\sigma_t^2\left(Z_t^2 - 1\right)$ as error we use local polynomial technique to estimate the functions $\phi_i(u), i = 0, 1, \cdots, p$ as described in [16]. Therefore, given a Kernel function $K(\cdot)$, we get the estimator at point $u_0$ by minimizing,

$$L = \sum_{j=p+1}^{n} \left( R_j^2 - \sum_{k=0}^{g} \left( \phi_{0k} + \sum_{i=1}^{p} \phi_{ik} R_{j-i}^2 \right) \left(u_j - u_0\right)^k \right)^2 K_{h_1}\left(u_j - u_0\right). \tag{38}$$

where $g$ is the polynomial degree, $\frac{t}{n} = u_t$, $K_{h_1}(.) = \left(\frac{1}{h_1}\right) K\left(\frac{.}{h_1}\right)$ and $h_1$ is the bandwidth.

Therefore, the initial $\sigma_t^2$ is given by,

$$\hat{\sigma}_t^2 = \hat{\phi}_0\left(\frac{t}{n}\right) + \sum_{i=1}^{p} \hat{\phi}_i\left(\frac{t}{n}\right) R_{t-i}^2. \tag{39}$$

**Step 2.** We use the conditional variance initially estimated in Step 1 to get the estimates of the time varying GARCH(p, q) parametric functions. We can write 35 as

$$\begin{aligned} R_t^2 &= w\left(\frac{t}{n}\right) + \sum_{i=1}^{p} \phi_i\left(\frac{t}{n}\right) R_{t-i}^2 + \sum_{i=1}^{q} \theta_i\left(\frac{t}{n}\right) \hat{\sigma}_{t-i}^2 \\ &\quad - \Theta\left(\frac{t}{n}\right)\left(\sum_{i=1}^{q}\left(\hat{\sigma}_{t-i}^2 - \sigma_{t-i}^2\right)\right) + \sigma_t^2\left(Z_t^2 - 1\right). \end{aligned} \tag{40}$$

Here, for a particular choice of bandwidth $E\left(\hat{\sigma}_t^2 - \sigma_t^2\right)$ is asymptotically negligible. Treating $\sigma_t^2\left(Z_t^2 - 1\right)$ as error we use local polynomial technique to estimate the functions $\phi_i(u), i = 0, 1, \cdots, p$ and $\theta_i(u), i = 0, 1, \cdots, q$ as described in [16]. The estimates at point $u_0$ are obtained by minimizing

$$L = \sum_{t=2}^{n}\left(R_t^2 - \sum_{k=0}^{g}\left(w_{k2} + \sum_{i=1}^{p} \phi_i R_{t-i}^2 + \sum_{j=1}^{q} \theta_j \sigma_{t-j}^2\right)\left(u_t - u_0\right)^k\right)^2 K_{h_2}\left(u_t - u_0\right). \tag{41}$$

where $g$ is the polynomial degree, $\frac{t}{n} = u_t$, $K_{h_2}(.) = \left(\frac{1}{h_2}\right) K\left(\frac{\cdot}{h_2}\right)$ and $h_2$ is the bandwidth.

The final estimates for $\sigma_t^2$ in time varying GARCH(p, q) is obtained using these estimates as follows

$$\hat{\sigma}_t^2 = \hat{w}\left(\frac{t}{n}\right) + \sum_{i=1}^{p} \hat{\phi}_i\left(\frac{t}{n}\right) R_{t-i}^2 + \sum_{j=1}^{q} \hat{\theta}_j\left(\frac{t}{n}\right) \sigma_{t-i}^2. \tag{42}$$

**Bandwidth Selection**

The bandwidth governs the complexity of the model, and therefore the choice of the smoothing parameter $h$ is of crucial importance for every kernel regression. Generally, we have to choose the bandwidth carefully to balance the bias and variance. A large bandwidth over-smooths the data resulting to underfitting, whereas a small bandwidth restricts neighbourhood size resulting in overfitting [17]. The data-driven bandwidth selection procedures most commonly used are, direct plug-in method, cross-validated bandwidth method, least-squares cross-validation method, smoothed cross-validation method, and the contrast method. Using the cross validation method based on the best linear predictor of $R_t^2$ given the past. The bandwidth $h$ is chosen for which

$$CV(h) = \frac{1}{n-1} \sum_{t=2}^{n}\left(R_t^2 - \hat{w}\left(\frac{t}{n}\right) + \sum_{i=1}^{p} \hat{\phi}_i\left(\frac{t}{n}\right) R_{t-i}^2 + \sum_{j=1}^{q} \hat{\theta}_j\left(\frac{t}{n}\right) \sigma_{t-i}^2\right), \tag{43}$$

is minimum, where the estimators are obtained through local polynomial technique by leaving the $t^{\text{th}}$ observation. The bandwidth selection procedure is computationally too cumbersome, especially when $n$ is large [16]. For a simplified version of cross validation method to reduce the computational complexity and make the bandwidth selection easy and doable refer to [16].

In random periodic case we apply a fixed value for bandwidth $h$ which is proportional to random periodicity $\tau$. That is,

$$h = \frac{\tau}{n}. \tag{44}$$

Therefore, we will only consider $\tau$ nearest neighbours of $u_0$ for each target covariate $u_0$.

**Kernel Function**

The kernel function choice $K(u)$ determines the local neighborhood's shape over which the smoothing is performed. The kernel function has the following properties:

1) $K(u) \geq 0, \forall u$

2) $\int_{-\infty}^{\infty} K(u)\mathrm{d}u = 1$

3) $K(-u) = K(u), \forall u$

The first two properties are those of probability density function and the third property implies that the Kernel density is symmetrical. The particular form of the function has only a relatively small effect on estimation accuracy. Hence, Gaussian kernel

$$K(u) = \frac{1}{(2\pi)^{0.5}} \mathrm{e}^{-\frac{1}{2}u^2}, \tag{45}$$

which is differentiable with low computational complexity is most commonly used.

## 5. Conclusions and Suggestions

Modelling complex periodic time series is quite tedious particularly when periodicity is mixed with randomness. The ANN Time varying GARCH model incorporates the modelling power of artificial neural networks and time varying GARCH models in order to model processes with fixed and random periodicity. The proposed model whose theoretical background in parameter estimation is developed in this paper through non parametric methods forms one of the candidates models in modelling complex financial time series. More research on the model properties and application with real data is still needed in order to entirely develop a comprehensive theoretical framework of the proposed model.

## Acknowledgements

We wish to thank the reviewers.

## Conflicts of Interest

The authors declare no conflict of interest.

## References

[1] Adhikari, R. and Agrawal, R. (2013) An Introductory Study on Time Series Modeling and Forecasting.

[2] Frances, P. (1996) Periodicity and Stochastic Trends in Economic Time Series. OUP Catalogue, Oxford University Press, Oxford.

[3] Zhang, G. (2003) Time Series Forecasting Using a Hybrid ARIMA and Neural

Network Model. *Neurocomputing*, **50**, 159-175. https://doi.org/10.1016/S0925-2312(01)00702-0

[4] Liu (2019) ARMA Model for Random Periodic Processes. Loughborough University, Loughborough.

[5] Feng and Zhao (2015) Random Periodic Processes, Periodic Measures and Ergodicity.

[6] Boger, Z. and Guterman, H. (1997) Knowledge Extraction from Artificial Neural Network Models. 1997 *IEEE International Conference on Systems*, *Man*, *and Cybernetics*. *Computational Cybernetics and Simulation*, Orlando, 12-15 October 1997, Vol. 4, 3030-3035.

[7] Kihoro, J., Otieno, R. and Wafula, C. (2004) Seasonal Time Series Forecasting: A Comparative Study of ARIMA and ANN Models. *African Journal of Science and Technology*, **5**, 41-49.

[8] Bishop Christopher, M., *et al.* (1995) Neural Networks for Pattern Recognition. Oxford University Press, Oxford.

[9] Bouwmans, T., Javed, S., *et al.* (2019) Deep Neural Network Concepts for Background Subtraction: A Systematic Review and Comparative Evaluation. *Neural Networks*, **117**, 8-66. https://doi.org/10.1016/j.neunet.2019.04.024

[10] Dahlhaus, R., Rao, S., *et al.* (2006) Statistical Inference for Time-Varying ARCH Processes. *Annals of Statistics*, **34**, 1075-1114. https://doi.org/10.1214/009053606000000227

[11] Xu, S. and Chen, L. (2008) A Novel Approach for Determining the Optimal Number of Hidden Layer Neurons for FNN's and Its Application in Data Mining.

[12] Blum, A. (1992) Neural Networks in C++ an Object-Oriented Framework for Building Connectionist Systems. John Wiley and Sons, Inc., Hoboken.

[13] Ash, T. (1989) Dynamic Node Creation in Back-Propagation Networks. *Connection Science*, **1**, 365-375. https://doi.org/10.1080/09540098908915647

[14] Hirose, Y., Yamashita, K. and Hijiya, S. (1991) Back-Propagation Algorithm Which Varies the Number of Hidden Units. *Neural Networks*, **1**, 61-66. https://doi.org/10.1016/0893-6080(91)90032-Z

[15] Rivaks, I. and Personnaz, L. (2000) A Statistical Procedure for Determining the Optimal Number of Hidden Neurons of a Neural Model. *Second International Symposium on Neural Computation* (*NC*'2000), Berlin, 23-26 May 2000, 14-17.

[16] Rohan, N. and Ramathan, T. (2013) Nonparametric Estimation of a Time-Varying GARCH Model. *Journal of Nonparametric Statistics*, **25**, 33-52. https://doi.org/10.1080/10485252.2012.728600

[17] Gajewicz, S., Kar, S. and Piotrowska, M. (2021) The Kernel-Weighted Local Polynomial Regression (KwLPR) Approach: An Efficient, Novel Tool for Development of QSAR/QSAAR Toxicity Extrapolation Models. *Journal of Cheminformatics*, **13**, 1-20. https://doi.org/10.1186/s13321-021-00484-5