Iranian Economic Review, Vol.16, No.30, Fall 2011

A Comparative Study of Parametric and Nonparametric Regressions

Shahram Fattahi[*]

Received: 2010/08/28 Accepted: 2011/04/24

Abstract

This paper evaluates inflation forecasts made by parametric and nonparametric models. The results revealed that the neural network model yields better estimates of inflation rate than do parametric autoregressive integrated moving average (ARIMA) and linear models. Furthermore, the neural network model outperformed nonparametric models (except MARS).

Keywords: ARIMA, AM, MARS, PPR, NN, Inflation Forecast.

1- Introduction

The parametric regression approach is based on the prior knowledge of the functional form relationship. If the knowledge is correct, the parametric method can model most data sets well. However, if the wrong functional form is chosen a priori, this will result in larger bias as compared to competitive models (Fan and Yao, 2003). Parametric linear models, as a type of parametric regression, are frequently used to describe the association between the dependent variable and explanatory variables. They require the estimation of a finite number of parameters. Furthermore, parametric linear dynamic models which are based on a theoretical or data-driven approach will be employed.

Why is nonparametric regression important? Over the last decade, increasing attention has been devoted to nonparametric regression as a new

[*]. Assistant Professor, Department of Economics, Razi University.

technique for estimation and forecasting in different sciences including economics. Nonparametric regression analysis relaxes the assumption of the linearity in the regression analysis and enables one to explore the data more flexibly. However, in high dimensions variance of the estimates rapidly increases, called as “curse of dimensionality”, due to the sparseness of data. To overcome this problem, some nonparametric methods have been proposed such as additive model (AM), multiple adaptive regression splines (MARS), projection-pursuit regression (PPR) and neural networks (NN).

This study compares parametric and nonparametric methods. The agents are assumed to use an optimal parametric autoregressive moving average (ARMA) model or nonparametric models including Additive model (AD), Multiple Adaptive Regression Splines (MARS), Projection-Pursuit Regression (PPR), and Neural Networks (NN) for forecasting. In fact, out-of-sample estimates of inflation generated by the parametric and nonparametric models will be compared.

The paper is divided into four sections. Following introduction in section 1 we discuss parametric and nonparametric methods in section 2. Section 3 reports the empirical results of this study. Finally, section 4 concludes.

2- Statistical predictors

2-1- Parametric prediction models

A brief review of ARIMA modeling is presented (Chatfield, 2000). Autoregressive integrated moving average (ARIMA) or (Box-Jenkins) models are the basis of many fundamental ideas in time-series analysis. In order to analyze a time series, it must be assumed that the structure of the stochastic process which generates the observations is essentially invariant through time. The important assumption is that of stationarity, which requires the process to be in a particular state of ‘statistical equilibrium’ (Box and Jenkins, 1976).

An autoregressive moving average process: ARMA (p,q) is obtained by combining p autoregressive terms and q moving average terms and can be written as

$$\phi(L)X_t = \alpha + \theta(L)\varepsilon_t$$

with AR polynomial $\phi(L) = 1 - \phi_1 L - \phi_2 L^2 - \ldots - \phi_p L^p$ and

MA polynomial $\theta(L) = 1 + \theta_1 L + \theta_2 L^2 + ... + \theta_q L^q$. An ARMA model is stationary provided that the roots of $\phi(L) = 0$ lie outside the unite circle. This process is invertible if the roots of $\theta(L) = 0$ lie outside the unite circle. Low order ARMA models are of much interest since many real data sets are well approximated by them rather than by a pure AR or pure MA model. In general, ARMA models need fewer parameters to describe the process.

In most cases economic time series are non-stationary and therefore we cannot apply ARMA models directly. One possible way to remove the problem is to take difference so as to make them stationary. Non-stationary series often become stationary after taking first difference ($X_t - X_{t-1} = (1-L)X_t$). If the original time series is differenced d times, then the model is said to be an ARIMA (*p*, *d*, *q*) where 'I' stands for integrated and *d* denotes the number of differences taken. Such a model is described by

$$\phi(L)(1-L)^d X_t = \alpha + \theta(L)\varepsilon_t$$

The combined AR operator is now $\phi(L)(1-L)^d$. The polynomials $\phi(z)$ and $\theta(z)$ have all their roots outside the unit circle. The model is called integrated of order d and the process is said to have d unit roots.

2-2- Nonparametric prediction models

2-2-1- Nonparametric Smoothers

The general nonparametric regression model (Fox, 2000, 2005) is as follows:

$$y_i = f(X_i^{'}) + \varepsilon_i$$

$$= f(x_{i1}, x_{i2}, ..., x_{ik}) + \varepsilon_i \qquad \varepsilon_i \sim NID(0, \sigma^2)$$

The regression function is $f(.)$ unspecified in advance and is estimated directly. In fact, there is no parameter to estimate. It is implicitly assumed that $f(.)$ is a smooth, continuous function. If there is only one predictor $y_i = f(x_i) + \varepsilon_i$ then it is called 'scatter plot smoothing' because it traces a smooth curve through a scatter plot of y against x.

There are several smoothers such as local averaging, kernel smoother, Weighted Scatterplot Smoothing (Lowess) and spline Smoother that fit a linear or polynomial regression to the data points in the vicinity of x and then use the smoothed value as the predicted value at x.

2-2-1-1- Local Averaging

In local averaging procedures, we move a window continuously over the data, averaging the observations that fall in the window. The estimated values $\hat{f}(x)$ at a number of focal values of x are calculated and connected. It is possible to use a window of fixed width or to adjust the width of window to include a constant number of observations. Local averages are usually subject to boundary bias, roughness and distortion (when outliers fall in the window).

2-2-1-2- Kernel Smoother

A Kernel smoother is an extension of local averaging and usually produces a smoother result. At the focal value x_0 , it is of the form

$$\hat{f}(x_0) = \sum_{i=1}^{n} y_i K\left(\frac{x_i - x_0}{b}\right) \Bigg/ \sum_{i=1}^{n} K\left(\frac{x_i - x_0}{b}\right)$$

where b is a bandwidth parameter, and K a kernel function. The Gaussian kernel $(K_N(z))$ and the tricube kernel $(K_T(z))$ are popular choices of kernel functions.

$$K_N(z) = \frac{1}{\sqrt{2\pi}} e^{-z^2/2}$$

$$K_T(z) = \begin{cases} (1\text{-}|z|^3)^3 & \text{for} \quad |z|<1 \\ 0 & \text{for} \quad |z|\geq 1 \end{cases}$$

For the Gaussian kernel the bandwidth b is the standard deviation of a normal distribution and for the tricube kernel b is the half-width of a window enclosing the observations for the local regression. Although the kernel smoother has a better performance as compared to the local average regression, it is still subject to boundary bias.

It is implicitly assumed that the bandwidth b is fixed, but it is possible for kernel smoothers to be adapted to nearest-neighbor bandwidths. We can adjust b (x) so that a constant number of observations m are included in the window. The fraction m/n is called the span of the kernel smoother and is chosen based on a cross-validation approach. The kernel estimator can produce smoother results using larger bandwidths. In fact, there is a direct relationship between the span and smoothing degree: the larger the span, the smoother the result.

2-2-1-3- Lowess Smoother

As mentioned above, the kernel estimation has some problems. Local polynomial regression tries to overcome these difficulties and provides a generally adequate method of nonparametric regression which extends to additive regression (Fox, 2005). An implementation of local polynomial regression is lowess (Cleveland, 1979). The algorithm used by lowess smoothing applies robust locally linear fits. It is similar to local averaging but the data points that lie in the window are weighted so that nearby points get the most weight and a robust weighted regression is used.

We can examine local polynomial regression in two cases: simple regression and multiple regression.

Simple Regression: suppose we want to estimate the simple regression $y_i = f(x_i) + \varepsilon_i$ at a particular x-value, for example x_0. Local polynomial regression extends kernel estimation to a polynomial fit at x_0, using local kernel weights, $w_i = K[(x_i - x_0)/b]$. We implement a pth-order weighted-least-squares polynomial regression of y on x,

$$y_i = \alpha + \beta_1(x_i - x_0) + \beta_2(x_i - x_0)^2 + ... + \beta_p(x_i - x_0)^p + e_i$$

to minimize the weighted residual sum of squares, $\sum_{i=1}^{n} w_i e_i^2$. This procedure is repeated for representative values of x. As in kernel regression, the bandwidth b can either be fixed or variable, b(x), and the span of the local-regression smoother is selected based on a cross-validation approach.

Multiple Regression: in this case, $y_i = f(X_i^{'}) + \varepsilon_i$, we need to define a a multivariate neighborhood around a focal point $x_0^{'} = (x_{01}, x_{02}, ..., x_{0k})$. Furthermore, Euclidean distance is employed in the lowess function as: $D(x_i, x_0) = \sqrt{\sum_{j-=1}^{k}(z_{ij} - z_{0j})^2}$ where the z_{ij} are the standardized predictors,

$z_{ij} = x_{ij} - \bar{x}_j / s_j$, $\bar{x}_j$ is the mean of the jth predictor and s_j is its standard deviation. Calculating weights are based on the scaled distances:

$$w_i = W\left[\frac{D(x_i, x_0)}{b}\right]$$

Where w (.) is a weight function. In some cases, b needs to be adjusted to define a neighborhood including the [ns] nearest neighbors of x_0 (where the square brackets denote rounding to the nearest integer).

As a simple example, a local linear fit takes the form:

$$y_i = \alpha + \beta_1(x_{i1} - x_{01}) + \beta_2(x_{i2} - x_{02})^2 + ... + \beta_k(x_{ik} - x_{0k}) + e_i$$

The combinations of predictor values are used repeatedly to create the regression surface.

2-2-1-4- Spline Smoother

Suppose we have n pairs (x_i, y_i). A smoothing spline equation is considered as

$$ss(h) = \sum_{i=1}^{n}\left[y_i - f(x_i)\right]^2 + h\int_{x_{\min}}^{x_{\max}}\left[f^{''}(x)\right]^2 dx$$

The equation consists of two terms. The first term is the residual sum of squares and the second term is a roughness penalty. The object is to find the function $\hat{f}(x)$ with two continuous derivatives that minimized the penalized sum of squares. Here h is a smoothing parameter. For h=0, $\hat{f}(x)$ will interpolate the data if the x_i are distinct; this is similar to a local-regression estimate with span=1/n. If h is very large, then $\hat{f}$ will be selected so that $\hat{f}^{''}(x)$ is everywhere 0, which implies globally linear least-squares fit to the data. This is again similar to local regression with infinite neighborhoods.

The Spline Smoother is more attractive than local regression because there is an explicit objective-function to optimize. But it is not easy to generalize splines to multiple regression. Generally, the smoothing parameter h is selected indirectly by setting the equivalent number of parameters for the smoother .Both smoothing-spline and local-regression fits with the same degree of freedom are usually very similar.

2-2-2- Nonparametric Models

2-2-2-1- Additive Model (AM)

Nonparametric regression based on kernel and smoothing spline estimates in high dimensions faces two problems, that is, the curse of dimensionality and interpretability. Stone (1985) proposed the additive model to overcome these problems. In this model, since each of the individual additive terms is estimated using a univariate smoother, the curse of dimensionality is avoided. Furthermore, while the nonparametric form makes the model more flexible, the additivity allows us to interpret the estimates of the individual terms. Hastie and Tibshirani (1990) proposed generalized additive models for a wide range of distribution families. These models allow the response variable distribution to be any member of the exponential family of distributions. We can apply additive models to Gaussian response data, logistic regression models for binary data, and loglinear or log-additive models for Poisson count data.

A generalized additive model has the form

$$Y = \alpha + f_1(X_1) + f_2(X_2) + ... + f_p(X_p) + \varepsilon$$

where $f_j(.)$ are unspecified smooth (partial-regression)functions. We fit each function using a scatterplot smoother and provide an algorithm for simultaneously estimating all j functions. Here an additive model is applied to a logistic regression model as a generalized additive model. Consider a logistic regression model for binary data. The mean of the binary response $\mu(X) = \Pr(Y = 1|X)$ is related to the explanatory variables via a linear regression model and the logit link functions:

$$\log\left(\frac{\mu(X)}{1-\mu(X)}\right)=\alpha+\beta_1 X_1+\ldots+\beta_j X_j$$

The additive logistic model replaces each linear term by a more general functional form

$$\log\left(\frac{\mu(X)}{1-\mu(X)}\right)=\alpha+f_1(X_1)+\ldots+f_j(X_j)$$

In general, the conditional mean $\mu(X)$ of a response Y is related to an additive function of the explanatory variables via a link function g:

$$g[\mu(X)]=\alpha+f_1(X_1)+\ldots+f_j(X_j)$$

The functions f_j are estimated in a flexible way using the backfitting algorithm. This algorithm fits an additive model using regression-type fitting mechanisms.

Consider the jth set of partial residuals

$$\varepsilon_j=Y-(\alpha+\sum_{k\neq j} f_k(X_k))$$

Then $E(\varepsilon_j \mid X_j)=f_j(X_j)$. This observation provides a way for estimating each $f_j(.)$ given estimates $\left\{\hat{f}_j(.), i\neq j\right\}$ for all the others. The iterative process is called the backfitting algorithm (Friedman and Stuetzle, 1981).

2-2-2-2- Multiple Adaptive Regression Splines (MARS)

This approach (Friedman (1991), Hastie et al (2001)) fits a weighted sum of multivariate spline basis functions and is well suited for high-dimensional problems, where the curse of dimensionality would likely create problems for other methods. The MARS uses the basis functions $(x-t)_+$ and $(t-x)_+$ in the following way

$$(x-t)_+=\begin{cases}\text{x-t} & \text{if} \quad \text{x>t}\\ 0 & \text{otherwise}\end{cases}$$

$$(t-x)_+=\begin{cases}\text{t-x} & \text{if} \quad \text{x<t}\\ 0 & \text{otherwise}\end{cases}$$

The "+" denotes positive part. Each function is piecewise linear or linear spline, with a knot at value t. These functions are called a reflected pair for each input X_j with knots at each observed value x_{ij} of that input, and then the set of basis functions is defined as

$$C = \left\{ (X_j - t)_+, (t - X_j)_+ \right\}$$

The strategy for model-building is a forward stepwise linear regression using functions from the set C and their products. Thus the MARS model has the form

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X)$$

where the coefficients β_m are estimated by minimizing the residual sum-of-squares and each $h_m(X)$ is a function in C. By setting $h_0(X) = 1$ (constant function), the other multivariate splines are products of univariate spline basis functions:

$$h_m(X) = \prod_{s=1}^{k_m} h(x_{i(s,m)} \left| t_{s,m} \right.) \qquad 1 \le m \le k$$

where the subscript $i(s,m)$ means a particular explanatory variable, and the basis spline in that variable has a knot at $t_{s,m}$. k_m is the level of interactions between $i(s,m)$ variables and the values of m, $k_1, k_2, ..., k_m$, are the knot sets. Explanatory variables in the model can be linearly or non-linearly and are chosen for inclusion adaptively from the data. The model will be additive if the order of interactions equals one ($k = 1$).

A backward deletion procedure is used in the MARS model to prevent overfitting. The basis functions which have little contributions to the accuracy of fit are deleted from the model at each stage, producing an estimated best model $\hat{f}(\lambda)$ of each size λ. We can apply a generalized cross-validation criterion to estimate the optimal value of λ in the following way

$$GCV(\lambda) = \frac{\sum_{i=1}^{N} (y_i - \hat{f}_\lambda(x_i))^2}{(1 - M(\lambda)/N)^2}$$

The value of M (λ) includes the number of basis functions and the number of parameters used in selecting the optimal positions of the knots.

2-2-2-3- Projection-Pursuit Regression (PPR)

If the explanatory vector X is of high dimension, the additive model does not cover the effect of interactions between the independent variables. Projection-Pursuit Regression (Friedman and Stuetzle, 1981) applies an additive model to projected variables, projecting predictor variables X in M, as follows

$$Y = \sum_{m=1}^{M} g_m(w_m^T X) + \varepsilon \qquad E(\varepsilon) = 0, \mathrm{var}(\varepsilon) = \sigma^2$$

where w_m are unit p-vectors of unknown parameters. The functions g_m are unspecified and estimated along with the direction w_m using some flexible smoothing method. The PPR model employs the backfitting algorithm and Gauss-Newton search to fit Y.

The functions $g_m(w_m^T X)$ are called the ridge functions because they are constant in all but one direction. They vary only in the direction defined by the vector w_m. The scalar variable $V_m = (w_m^T X)$ is the projection X onto the unit vector w_m. The aim is to find w_m to yield the best fit to the data. If M is chosen large enough then the PPR model can approximate arbitrary continuous function of X (Diaconis and Shahshahani, 1984). However, in this case there is a problem of interpretation of the fitted model since each input enters into the model in a complex and multifaceted way (Hastie et al, 2001). As a result, the PPR model is a good option only for forecasting.

To fit a PPR model, we need to minimize the error function

$$E = \sum_{i=1}^{N} [y_i - \sum_{m=1}^{M} g_m(w_m^T x_i)]^2$$

over functions g_m and direction vectors w_m. The g and w are estimated by iteration. Imposing complexity constraints on the g_m is needed to avoid overfitting. There are two stages to estimate g and w. First, to obtain an estimate of g, suppose there is one term (M=1). We can form the derived variables $v_i = w^T x_i$ for any value of w. This implies a one-dimensional smoothing problem and any scatterplot smoother such as smoothing spline can be used to estimate g. Second, we minimize E over w for any value of g. These two steps are iterated until convergence. If there is more than one term in the PPR model then the model is built in a forward stage-wise manner that at each stage a pair (w_m, g_m) is added.

2-2-2-4- Neural Networks

Many recent methods to developing data-driven models have been inspired by the learning abilities of biological systems. For instance, most adults drive a car without knowledge of the underlying laws of physics and humans as well as animals can recognize patterns for the tasks such as face, voice or smell recognition. They learn them only through data-driven interaction with the environment. The field of pattern recognition considers such abilities and tries to build artificial pattern recognition systems that can imitate human brain. The interest to such systems led to extensive studies about neural networks in the mid-1980s (Cherkassky and Mulier, 2007).

Why use Neural Networks? Neural network modeling has seen an explosion of interest as a new technique for estimation and forecasting in economics over the last decades. They are able to learn from experience in order to improve their performance and to adapt themselves to changes in the environment. In fact, they can derive trends and detect patterns from complicated or imprecise data, and then model complex relationships between explanatory variables (inputs) and dependent variables (outputs). They are resistance to noisy data due to a massively parallel distributed processing.

Learning in neural network model

Stochastic approximation (or gradient descent) is one of the basic nonlinear optimization strategies commonly used in statistical and neural network methods (Cherkassky and Mulier, 2007). The gradient-descent methods are based on the first –order Taylor expansion of a risk functional

$$R(w) = \int L(y, f(x, w)) p(x, y) dxdy \qquad (1)$$

where $R(w)$ is the risk functional, $L(y, f(x, w))$ the loss function and $p(x, y)$ the joint probability density function. For regression, a common loss function is the squared error

$$L(y, f(x,w)) = (y - f(x,w))^2 \tag{2}$$

Learning is then defined as the process of estimating the function $f(x, w_0)$ that minimizes the risk functional

$$R(w) = \int (y - f(x,w))^2 p(x,y)dxdy$$

using only the training data. Although the gradient-descent methods are computationally rather slow, their simplicity has made them popular in neural networks. We will examine two cases to describe such methods: linear parameter estimation and nonlinear parameter estimation.

Linear Parameter Estimation

Consider a linear (in parameters) approximating function and the loss function specified above. For the task of regression, it can be shown that the empirical risk is as follows

$$R_{emp}(w) = \frac{1}{n}\sum_{i=1}^{n} L(x_i, y_i, w) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i, w))^2 \tag{3}$$

This function is to be minimized with respect to the vector of parameters w. Here the approximating function is a linear combination of fixed basis functions

$$\hat{y} = f(x,w) = \sum_{j=1}^{m} w_j g_j(x) \tag{4}$$

For some (fixed) **m**. The updating equation for minimizing $R_{emp}(w)$ with respect to w is

$$w(k+1) = w(k) - \gamma_k \frac{\partial}{\partial w} L(x(k), y(k), w) \tag{5}$$

where $x(k)$ and $y(k)$ are the sequences of input and output data samples presented at iteration step k. The gradient above can be written as

$$\frac{\partial}{\partial w_j} L(x,y,w) = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_j} = 2(\hat{y} - y) g_j(x) \qquad (6)$$

Now the local minimum of the empirical risk can be computed using the gradient (6). Let us start with some initial values $w(0)$. The stochastic approximation method for parameter updating during each presentation of *k*th training sample is as:

Step 1: Forward pass computations.

$$z_j(k) = g_j(x(k)), \qquad 1,..,\mathrm{m} \qquad (7)$$

$$\hat{y}(k) = \sum_{j=1}^{m} w_j(k) z_j(k) \cdot \qquad (8)$$

Step 2: Backward pass computations.

$$\delta(k) = \hat{y}(k) - y(k) \qquad (9)$$

$$w_j(k+1) = w_j(k) - \gamma_k \delta(k) z_j(k) \qquad (10)$$

where the learning rate γ_k is a small positive number decreasing with k. In the forward pass, the output of the approximating function is computed whereas in the backward pass, the error term (9), which is called “delta” in neural network literature, for the presented sample is calculated and utilized to modify the parameters. The parameter updating equation (10), known as delta rule, updates parameters with every training sample.

Nonlinear Parameter Estimation

The standard method used in the neural network literature is the back propagation algorithm which is an example of stochastic approximation strategy for nonlinear approximating functions. As it was considered already, the mapping from inputs to output given by a single layer of hidden units is as follows

$$f(x,w,V) = w_0 + \sum_{j=1}^{n} w_j g(v_{0j} + \sum_{i=1}^{d} x_i v_{ij}) \tag{11}$$

In contrast to (4), the set of functions is nonlinear in the parameters V. We seek values for the unknown parameters (weights) V and w that make the model fit the training data well. To do so, the sum of squared errors as a measure of fit must be minimized:

$$R_{emp} = \sum_{i=1}^{n} (f(x_i, w, V) - y_i)^2 \tag{12}$$

The stochastic approximation procedure for minimizing R_{emp} with respect to the parameters V and w is

$$V(k+1) = V(k) - \gamma_k \nabla_V L(x(k), y(k), V(k), w(k)), \tag{13}$$

$$w(k+1) = w(k) - \gamma_k \nabla_w L(x(k), y(k), V(k), w(k)), k = 1, ..., n, \tag{14}$$

where $x(k)$ and $y(k)$ are the *k*th training samples, presented at iteration step *k*. The loss function L is

$$L(x(k), y(k), V(k), w(k)) = \frac{1}{2}(f(x,w,V) - y)^2 \tag{15}$$

where the factor ½ is included only for simplifying gradient calculations in the learning algorithm. We need to decompose the approximation function (11) for computations of the gradient of loss function (15) as follows

$$a_j = \sum_{i=0}^{d} x_i v_{ij} \tag{16}$$

$$z_j = g(a_j), \tag{17}$$

$$z_0 = 1,$$

$$\hat{y} = \sum_{j=0}^{m} w_j z_j \tag{18}$$

For simplicity, we drop the iteration step *k*, consider calculation/ parameter update for one sample at a time and incorporate the terms w_0 and v_{0j} into the summations ($x_0 \equiv 1$). The relevant gradients, based on the chain rule of derivatives, are

$$\frac{\partial R}{\partial v_{ij}} = \frac{\partial R}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a_j} \frac{\partial a_j}{\partial v_{ij}}, \tag{19}$$

$$\frac{\partial R}{\partial w_j} = \frac{\partial R}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_j}. \tag{20}$$

In order to calculate each of the partial derivatives, we need equations (15) to (18). Therefore,

$$\frac{\partial R}{\partial \hat{y}} = \hat{y} - y \tag{21}$$

$$\frac{\partial \hat{y}}{\partial a_j} = g^{'}(a_j) w_j \tag{22}$$

$$\frac{\partial a_j}{\partial v_{ij}} = x_i \tag{23}$$

$$\frac{\partial \hat{y}}{\partial w_j} = z_j \tag{24}$$

If we plug the partial derivatives (21)-(24) into (19) and (20), the gradient equations are

$$\frac{\partial R}{\partial v_{ij}} = (\hat{y} - y) g^{'}(a_j) w_j x_i \tag{25}$$

$$\frac{\partial R}{\partial w_j} = (\hat{y} - y) z_j \tag{26}$$

Using these gradients and the updating equations, we can construct a computational method to minimize the empirical risk. Starting with some initial values w (0) and V (0), the stochastic approximation method updates weights upon presentation of a sample (x (k), y (k)) at iteration step k with learning rate γ_k as

- Step 1: Forward pass computations.

"Hidden layer"

$$a_j(k) = \sum_{i=0}^{d} x_i(k) v_{ij}(k), \tag{27}$$

$$z_j(k) = g(a_j(k)), \tag{28}$$

$$z_0(k) = 1$$

"Output layer"

$$\hat{y}(k) = \sum_{j=0}^{m} w_j(k) z_j(k) \cdot \tag{29}$$

- Step 2: Backward pass computations.

"Output layer"

$$\delta_0(k) = \hat{y}(k) - y(k) \tag{30}$$

$$w_j(k+1) = w_j(k) - \gamma_k \delta_0(k) z_j(k) \tag{31}$$

"Hidden layer"

$$\delta_{1j}(k) = \delta_0(k) g^{'}(a_j(k)) w_j(k+1) \tag{32}$$

$$v_{ij}(k+1) = v_{ij}(k) - \gamma_k \delta_{1j}(k) x_i(k) \tag{33}$$

In the forward pass, the output of the approximating function is computed whereas in the backward pass, the error term for the presented sample is calculated and utilized to modify the parameters in the output layer. Since it is possible to propagate the error at the output back to an error at each of the internal nodes a_j through the chain rule of derivatives, the procedure is called *error backpropagation*. In fact it is a propagation of the error signals from the output layer to the input layer.

The updating steps for output layer are similar to those for the linear case. Besides, the updating rule for the hidden layer is the same as the linear one but for the delta term (32). For this reason, backpropagation update rules (32) and (33) are usually called the “generalized delta rule”. The parameter updating algorithm holds if the sample size is large (infinite). However, if the number of training samples is finite, the asymptotic conditions of stochastic approximation are (approximately) satisfied by the repeated presentation of the finite training sample to the training algorithm. This is called recycling and the number of such repeated training samples is called the number of cycles (or epochs).

It is possible to use the backpropagation algorithm for networks with several output layers and networks with several hidden layers. For instance, if additional layers are added to the approximation function, then errors are ‘propagated’ from layer to layer by repeated application of generalized delta rule.

It should be noted that a neural network model can be identified as a pursuit projection regression (PPR) model (Hastie et al, 2001). In fact, the neural network with one hidden layer has the exactly the same form as the PPR model. The only difference is that the PPR model uses nonparametric functions ($g_m(v)$) while the neural network employs a simpler function which is based on a sigmoid transfer function.

3- Empirical Results

3-1- Background

The Iranian economy is oil-reliant so that any change in oil price can directly affect all economic sectors. It should be noted that Iran ranks second in the world in natural gas reserves and third in oil reserves. It is also OPEC’s second largest oil exporter. The economic sectors include the services sector, industry (oil, mining and manufacturing) and the agricultural sector. During the recent decades, the services sector has contributed the largest percentage of the GNP, followed by industry and agricultural sectors. The share of the services sector was 51 percent of GNP in 2003 while those of the industry and agricultural sectors were 35.1 and 13.9 percent of GNP respectively.

The Iranian economy has experienced a relatively high inflation averaging about 15 percent per year. The inflation rate has even been more than 21 percent on average after the 1973 oil crisis. Furthermore, there is a general agreement over the underestimating of the measured inflation due to price controls and government subsidies.

The empirical evidence implies that inflation is persistent in Iran. In other words, the effects of a shock to inflation result in a changed level of inflation for an extended period. To see this, the inflation rate is regressed on its own lags.

$$rgnpi_t = 0.44 rgnpi_{t-1} + 0.49 rgnpi_{t-2}$$

(t-value) (3.15) (3.42)

As the sum of coefficients on lagged inflation (0.93) is close to one, shocks to inflation have long-lasting effects on inflation.

3-2- Results

This study compares the performance of parametric and nonparametric regressions in the Iranian economy over the period 1959-2003. Different models will be applied to the inflation series and then the predicted values will be evaluated. In fact, out-of-sample estimates of inflation generated by the parametric and nonparametric models will be compared.

The empirical results of the Augmented Dicky-Fuller test indicated that all the variables employed are stationary, and thus this issue helps us to avoid the problem of the spurious relationships (see Appendix for the data source and definitions).

It is assumed agents use the lagged values of inflation and real GNP growth to forecast inflation. Figure 1.a and Figure 1.b demonstrate a local linear regression fit of inflation rate (rgnpi), defined as the rate of change of GNP deflator, on the lagged inflation rate (rgnpilag1) and lagged real GNP growth rate (rgnplag1) using the Lowess function for a variety of spans. If the fitted regression looks too rough, then we try to increase the span but if it looks smooth, then we will examine whether the span can be decreased without making the fit too rough (Fox, 2005). The objective is to find the smallest value of span (s) that provides a smooth fit. A trial and error procedure suggests that the span s=0.5 is suitable and it seems to provide a reasonable compromise between smoothness and fidelity to the data.

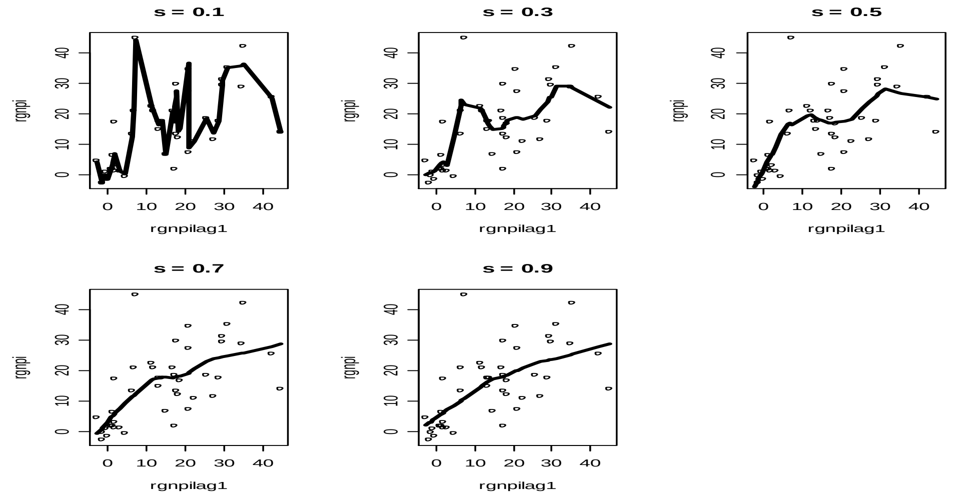

Figure 1-a: Local Linear Regression fit of Inflation Rate (Rgnpi) on the Lagged Inflation Rate (Rgnpilag1) Using Lowess Function for a Variety of Spans

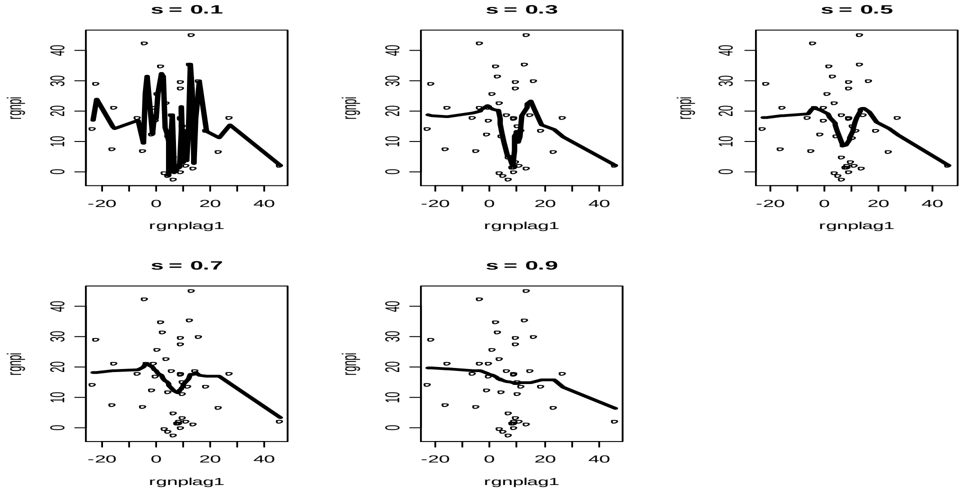

Figure 1-b: Local linear regression fit of inflation rate (rgnpi) on the lagged real GNP growth rate (rgnplag1) using Lowess function for a variety of spans

A test of nonlinearity is performed by contrasting the nonparametric regression model with the linear simple-regression model. We regress inflation on rgnpilag1 (Case 1) and rgnplag1 (Case2) separately. As a linear model is a special case of a nonlinear model, two models are nested. An F-test is formulated by comparing alternative nested models. The results is as follows

Linear model vs Nonparametric regression (Case1): F=8.78(p-value=0.008)
Linear model vs Nonparametric regression (Case2): F=6.48(p-value=0.04)

It is obvious that the relationship between the dependent variable and explanatory variables are significantly nonlinear. It should be noted that the variable rgnplags1 will not be significant if a linear regression is considered.

Since nonparametric regression based on smoothing functions faces the curse of dimensionality, the additive model has been proposed (Stone, 1985). The result of fitting an additive model using Lowess smoother can be written as

$$rgnpi = S(\text{rgnpilag1}) + S(\text{rgnplag1})$$

F	(4.13)	(4.43)
p-value	(0.01)	(0.03)

where S denotes the Lowess smoother function. It is obvious that both smoothers are significantly meaningful. Furthermore, the linear model is nested by the additive model with p-value being equal to 0.01. Figure 2 illustrates plots of the estimated partial-regression functions for the additive regression model. The points in each graph are partial residuals for the corresponding explanatory variable, removing the effect of the other explanatory variable. The broken lines demonstrate pointwise 95-percent confidence envelopes for the partial fits.

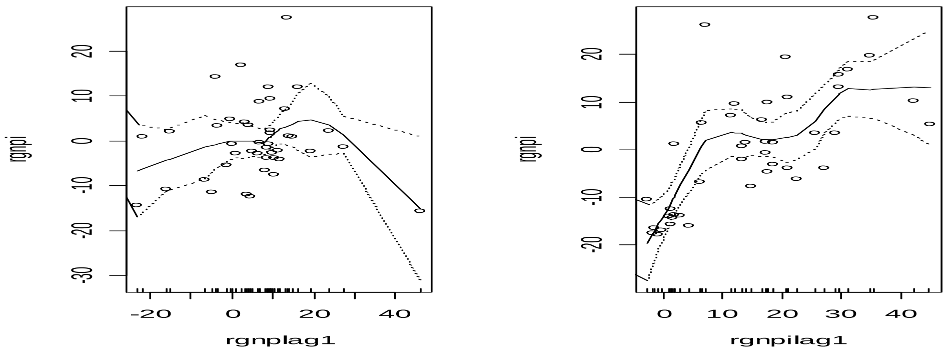

Figure 2: Plots of the Estimated Partial-Regression Functions for the Additive Regression of the Inflation Rate (Rgnpi) on the Lagged Real GNP growth Rate (rgnplag1) and the Lagged Inflation Rate (Rgnpilag1)

We use MARS model to fit a piecewise linear model with additive terms to the data. The results indicate that pairwise interaction terms (by degree=2 and degree=3) make little difference to the effectiveness of explanatory variables.

The additive model seems to be too flexible and it is not able to cover the effect of interactions between explanatory variables. To remove this problem, the Projection- Pursuit Regression model has been proposed. The PPR model applies an additive model to projected variables (Friedman and Stuetzle, 1981). Figure 3 shows plots of the ridge functions for the three two-term projection pursuit regressions fitted to the data.

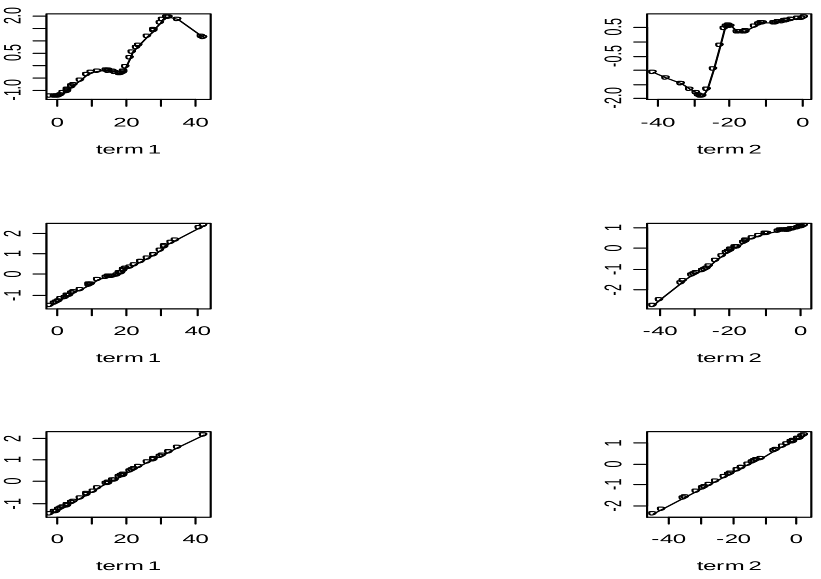

Figure 3: Plots of the Ridge regression for three two-term Projection Pursuit Regressions Fitted to the Data.

Although MARS model is an accurate method, it is sensitive to concurvity. Neural networks do not share this problem and are better able to predict in this situation. In fact, as neural networks are nonlinear projection methods and tend to overparameterize, they are not subject to concurvity (Hastie et al, 2001). We examined several neural network models and the results indicate that a 2-3-1 network has a better performance.

The Wilcoxon test has been used to compare the squared error of a neural network model and a rival model. In fact, out of sample forecasting has been employed to evaluate different models. The performance of PPR and additive models appears to differ from the neural network model, implying that the NN model can significantly outperform the PPR model and it has a better performance than the additive model, but not by much. Furthermore, the NN model is significantly better than the linear model (LM). However, there is no possibility that the NN model can outperform the MARS model (see Table 1).

Table 1: Out of Sample Forecasting based on Wilcoxon test

	p-value
PPR vs. NN	0.01
LM vs. NN	0.00
MARS vs. NN	1
AM vs. NN	0.38

Now we compare the NN model to the parametric autoregressive moving average (ARMA) model for inflation. A collection of ARMA (p, q) models, for different orders of p and q, have been estimated and then the best model was selected according to the Akaike information criterion (AIC) and the Schwarz information criterion (SIC). Examining the ARMA models for the inflation series indicated that ARMA (1, 1) is the best-fitting model.

Diagnostic checking, the correlogram (autocorrelations) of inflation from the regression tests was examined and confirmed the results. The last 5 observations are used for comparing the ex post forecasts generated by the two models. Furthermore, the Root Mean Square Error (RMSE) is used to evaluate ex post forecasts. We apply the feed-forward backpropagation as learning algorithm where only lagged inflation is used as input. The results imply that the forecasting performance of the NN model (RMSE=0.05) is significantly better than that of the ARMA model (RMSE=11.73). It should be noted that the results from the inflation lags exceeding one and more number of hidden layers are almost the same. Therefore, the NN model outperforms the parametric ARMA model.

3- Conclusions

This study examined the forecast performance of parametric and nonparametric models. The agents were assumed to use a parametric autoregressive moving average (ARMA) model or nonparametric models to forecast inflation. The results revealed that the neural network model yields better estimates of inflation rate than do parametric autoregressive moving average (ARMA) and linear models. Comparing to the nonparametric alternatives, the results of Wilcoxon tests demonstrated that the forecasting performance of Projection-Pursuit Regression and Additive models appeared to differ from the Neural Network model, implying that the Neural Network model can significantly outperform Projection-Pursuit Regression and Additive models. However, there was no possibility that the Neural Network model can outperform the Multiple Adaptive Regression Splines model.

References

1- Barucci E and Landi L(1998), Nonlinear Versus Linear Devices: A Procedural Perspective, Computational Economics, Vol. 12, pp. 171-191

2- Box, G.E.P and Jenkins, G (1976) Time Series Analysis: Forecasting and Control. San Francisco: Holden Day.

3- Chatfield C (2000), Time-Series Forecasting, Chapman & Hall/CRC

4- Cherkassky, V and Mulier F (2007), Learning from Data: Concepts, Theory, and Methods, John Wiley & Sons, Second Edition

5- Cleveland, W.S. (1979), Robust Locally Weighted Regression and Smoothing Scatterplots, Journal of the American Statistical Association 74 (368): 829–836.

6- Diaconis P and Shahshahani M (1984), On nonlinear functions of linear combinations, SIAM Journal of Scientific and Statistical Computing, Vol. 5, No. 1, pp. 175-191

7- Fan, J, Yao, Q (2003), Nonlinear Time Series: Nonparametric and Parametric Methods, Springer-Verlag: Berlin Heidelberg and New York.

8- Fox, J (2000), Multivariate Generalized Nonparametric Regression, Sage Publications.

9- Fox, J (2005), Introduction to Nonparametric Regression, McMaster University, Canada.

10- Friedman, JH, Stuetzle, W (1981), Projection Pursuit Regression, Journal of the American Statistical Association, Vol. 76, No. 376, pp. 817-823.

11- Friedman, HF (1991), Multivariate Adaptive Regression Splines, The Annals of Statistics, Vol. 19, No. 1, pp. 1-67.

12- Hastie, TJ, Tibshirani, RJ (1990), Generalized Additive Models, Chapman & Hall.

13- Hastie, TJ, Tibshirani, R, Friedman, J (2001), The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer-Verlag: Berlin Heidelberg and New York.

14- Stone, C.J. (1985), Additive Regression and Other Nonparametric Models, Annals of Statistics, 13, 689-705.

Appendix: Data source and definitions

The data are annually for the period 1959-2003 and are collected from the Central Bank of Iran.

gnp = real GNP (at the constant 1997 prices)

gnpi = GNP deflator (1997=100)