2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

A Higher-Order Fuzzy Neural Network for Modeling Financial Time Series

Massimo Panella, Luca Liparulo, and Andrea Proietti

Abstract— **This work investigates on the widespread use of fuzzy neural networks in time series forecasting, concerning in particular the energy commodity markets. We propose a new learning strategy suited to any neural model. The proposed approach is further assessed in the case of higher-order Sugeno-type fuzzy rules, which are able to replicate the daily data and to reproduce the same statistical features for various Commodity time series. The data used are obtained from the daily return series of specific energy commodities, such as coal, natural gas, crude oil and electricity, over the period 2001-2010 for both the European and US markets. We will prove that our approach can obtain interesting results in terms of prediction accuracy and volatility estimation, compared to well-known neural and fuzzy neural models and to the ARMA-GARCH statistical paradigm.**

I. Introduction

THIS paper proposes a new neural network approach for prediction and modeling of financial time series, focusing on the study of energy commodity market prices. This is a well-known challenging problem, given the non-stationarity and, often, non-linearity of time series, which results in a complex dynamic for prices that is hard to be successfully modeled by using standard econometric models.

Energy price dynamics are affected by complex risk factors, such as political events, extreme weather conditions and financial market behavior. Over the last 10 years the global demand for crude oil and gas has increased, largely due to the rapidly increasing demands of non OECD countries, especially China [1]. Local gas and coal are mainly used in the electricity generation process and recently their supply and demand experienced a profound transformation. The economic exploitation at higher prices of shale gas and shale oil is modifying the demand for fossil fuels.

It is critical to forecast the price direction of energy commodities in order to reduce the negative impact of high price fluctuations on investment results and on risk management strategies. Commodity prices forecasting on a daily basis cannot be easily obtained using standard structural models, given the lack of daily data on supply and demand, normally available monthly and a quarter in arrears. Reduced form models are commonly used to price energy commodities, i.e., two state variable stochastic models for oil and gas price dynamics [2], Markov regime switching models for electricity prices [3]. In most cases, energy price models deal with the solution of complex stochastic differential equations.

M. Panella, L. Liparulo, and A. Proietti are with the Department of Information Engineering, Electronics and Telecommunications (DIET), University of Rome "La Sapienza", Via Eudossiana 18, 00184 Rome, Italy (email: {massimo.panella, luca.liparulo, andrea.proietti}@uniroma1.it).

Various approaches have been proposed so far, ranging from standard econometric models to the ones based on computational intelligence. Some applications focus on the principal processes generating the observed time series and make use of neural networks as nonlinear models that are more suited to identify the chaotic behavior of specific commodity prices with respect to common Autoregressive Integrated Moving Average (ARIMA) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models.

Neural Networks are used as nonlinear regression models, which can generalize the stationary and univariate models used in econometrics in order to provide an effective tool for capturing the most important features we expect in Commodity price returns i.e., fat tails, volatility clustering or persistence and leverage effects [4]–[7]. Among these approaches well-known solutions were based, for example, on: hybrid methodology combining ARIMA models, neural networks and Web mining, from which to extract a rule-based system and non-linear integration; Generalized Pattern Matching based on Genetic Algorithm (GPMGA) for multi-step forecasting; Support Vector Machines (SVM); Multi-layer Perceptron (MLP), Elman Recurrent Neural Network (ERNN), and the Recurrent Fuzzy Neural Network (RFNN). By using a 'black-box' approach, the noisy information contained in the input data is exploited and some critical assumptions, often necessary to setup analytical models, are avoided. In a time of unique Commodity market dynamics and using a new approach to neural networks, we are intended to produce very robust predictions that can be used for both investment and risk management.

The dynamics of energy prices is actually complex and it has shown large unexpected volatility in the last decade. In this context, a powerful tool providing accurate price forecasting is needed. Natural gas, coal and electricity prices, unlike crude oil, present seasonality features that are usually measured using deterministic techniques. In this paper we aim to forecast short term price dynamics in order to be able to adequately measure the existing correlations between the various commodities. To this extent the seasonality component of the gas and coal prices will not affect the results. We apply the proposed approach to forecast crude oil, natural gas, electricity and coal prices using data of both European and US markets collected for the last decade.

The contribution of this paper is twofold. On one hand, we propose a framework for the prediction of expected values of times series and the related volatilities involving any regression approach. On the other hand, a new fuzzy neural model is proposed, where the consequent, output part of each

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

rule is generalized. The parameters of each rule are obtained through a clustering synthesis in the joint input-output space and a computationally efficient optimization is also proposed. It is based on a constructive procedure, by which the number of rules is progressively increased and the optimal one is automatically determined on the basis of learning theory in order to maximize the generalization capability of the resulting neural network. Extensive computer simulations prove the validity of the proposed algorithm and show a favorable comparison with other well-established techniques.

II. A Generalized Neural Network Approach to Time Series Forecasting

Prices related to financial assets are commonly collected as time series uniformly sampled at hourly, daily or monthly rate. Let S_t be a generic time series of prices, where t denotes the time index. In order to work with stable data sets for modeling, the financial analysis is usually carried out on the return series y_t defined as:

$$y_t = \ln \frac{S_t}{S_{t-1}} \,. \tag{1}$$

Given the randomness of prices, a return series is conveniently modeled as a discrete time stochastic process, for which we wish to know the conditional probability density function (pdf) given the conditioning set $\mathcal{I}_{t-1}$ associated with all the available information available *prior to* time t (i.e., past observations and estimated models).

Almost all the prediction techniques aim to estimate the conditional moments of this pdf, which imply explicit dependence on past observations. We will assume that all the necessary estimations can be realized in a time smaller than the interval between two consecutive observations, in such a way we can limit our analysis to a 'one-step-ahead' prediction, for which the information $\mathcal{I}_{t-1}$ is available for the analysis prior to time t. Otherwise, the prediction should start earlier, by using the information $\mathcal{I}_{t-s}$, $s > 1$, to perform a prediction at distance s of the sample at time t.

As the reference background of econometrics, we consider in this work an additive model of time series:

$$y_t = \mu_t + \epsilon_t \,, \tag{2}$$

where μ_t is a deterministic component, representing the forecast, and ϵ_t is a random variable, which takes into account the uncertainty of prediction. In fact, $\epsilon_t = y_t - \mu_t$ can be considered as the forecast error, or 'innovation', and it is itself a random process usually assumed to having a zero-mean normal conditional distribution. Thus, the conditional mean of y_t is $\mathbb{E}_{y_t|\mathcal{I}_{t-1}}[y_t] = \mu_t$, $\mathbb{E}_{\epsilon_t|\mathcal{I}_{t-1}}[\epsilon_t] = 0$ and $\mathrm{Var}_{\epsilon_t|\mathcal{I}_{t-1}}(\epsilon_t) = \mathbb{E}_{\epsilon_t|\mathcal{I}_{t-1}}[\epsilon_t^2] = \mathrm{Var}_{y_t|\mathcal{I}_{t-1}}(y_t) = \sigma_t^2$.

The conditional variance σ_t^2 of y_t can be considered as the volatility of the return series at time t and its accurate estimation, with respect to the implied volatility of the actual process, is mandatory to any financial tool. In general, σ_t^2 changes over time and this phenomenon is called heteroskedasticity. Many models tends to capture this behavior, especially the GARCH model that is widely used in both theoretical and practical applications. A GARCH(P, Q) model is based on the following estimation of σ_t^2 by using past observations [8]:

$$\begin{aligned} &\sigma_t^2 = \alpha_0 + \sum_{i=1}^{Q} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{P} \beta_j \sigma_{t-j}^2 \,, \\ &P \geq 0 \,, \;\; Q > 0 \,, \\ &\alpha_0 > 0 \,, \;\; \alpha_i \geq 0 \,, \;\; i = 1 \ldots Q \,, \\ &\beta_j \geq 0 \,, \;\; j = 1 \ldots P \,. \end{aligned} \tag{3}$$

A common assumption when modeling financial time series is that forecast errors are zero-mean random disturbances that are serially uncorrelated from one period to the next although not independent, evidently.

A GARCH(P, Q) model is a generalization of the early ARCH model proposed by Engle in [9]; i.e., an ARCH(Q) model coincides with a GARCH($0, Q$) model. However, specifying the order P, Q of a GARCH model is not easy and it is still an open problem. Consequently, only low orders are usually adopted in most applications. Nevertheless, several extensions of the original GARCH model have been proposed in the past, by specifying a different parametrization to capture serial dependence on the conditional variance.

The time series model should be completed by a suited hypothesis of the conditional mean μ_t as well. A general choice can be based on the linear regression model:

$$\mu_t = a_0 + \sum_{i=1}^{R} a_i y_{t-i} + \sum_{j=1}^{M} b_j \epsilon_{t-j} \,. \tag{4}$$

Bearing in mind (2), y_t follows a general ARMA process where usual conditions are given on the eigenvalues associated with the characteristic AR and MA polynomials, in order to ensure stationarity and invertibility, respectively. Energy commodity returns are typically modeled as wide-sense stationary (WSS) processes, with constant unconditional mean and constant unconditional variance but non-constant conditional variance. In the following, we will consider a default association of an ARMA(R, M) process coupled with a GARCH(P, Q) WSS process, denoted as ARMA(R, M)-GARCH(P, Q), for which the unconditional variance $\mathrm{Var}_{y_t}(y_t)$ is a computable function of $\mathrm{Var}_{\epsilon_t}(\epsilon_t)$.

Some generalizations of the ARMA model are also possible for modeling the conditional mean μ_t; for example, the ARIMA model, which is stationarized by adding lags of the differenced series and/or lags of the forecast errors, and ARMA with eXogenous inputs (ARMAX), where some 'exogenous' or independent variables are added as an explanatory regression data. There are also nonlinear variants to such models as the Nonlinear ARMA (NARMA) and Nonlinear ARMAX (NARMAX). Finally, we outline that the previous ones are all univariate models. A multivariate generalization is possible by using Vector Autoregression (VAR) models, which are intended to capture the evolution and the interdependencies between multiple time series.

The general approach to time series modeling and prediction described so far evidences how both conditional mean μ_t

and volatility σ_t^2 can be estimated through a suited regression problem, which can be compactly defined by the following equation for μ_t:

$$\begin{cases} \mu_t = h_\mu \left(\mathbf{x}_t^{(\mu)}; \boldsymbol{\omega}_t^{(\mu)} \right) , \\ \mathbf{x}_t^{(\mu)} = \left[y_{t-1} \; y_{t-2} \; \dots \; y_{t-R} \; \epsilon_{t-1} \; \epsilon_{t-2} \; \dots \; \epsilon_{t-M} \right] , \end{cases} \tag{5}$$

and by the following equation for σ_t^2:

$$\begin{cases} \sigma_t^2 = h_\sigma \left(\mathbf{x}_t^{(\sigma)}; \boldsymbol{\omega}_t^{(\sigma)} \right) , \\ \mathbf{x}_t^{(\sigma)} = \left[\sigma_{t-1}^2 \; \sigma_{t-2}^2 \; \dots \; \sigma_{t-P}^2 \; \epsilon_{t-1}^2 \; \epsilon_{t-2}^2 \; \dots \; \epsilon_{t-Q}^2 \right] , \end{cases} \tag{6}$$

where the orders R, M, P, and Q are fixed in advance (they are analogous to ARMA and GARCH models); $\boldsymbol{\omega}_t^{(\mu)}$ and $\boldsymbol{\omega}_t^{(\sigma)}$ are the parameter vectors of the regression functions h_μ and h_σ, respectively, which obviously change over time.

By this approach, the dynamics of time series is modeled directly, as actually pursued by GARCH models. However, both linear and nonlinear ARMA-GARCH models are global regression methods that do not involve a parametric basis function expansion of the regression models, similarly to spline functions or the MLP neural network. For this reason they can be affected by the 'curse of dimensionality' problem, since their performances dramatically decrease with the increment of the model order because of the increasing sparseness of data in the input space. We propose the use of clustering for partitioning the data space, so that clusters of points in significant regions of the data space can be linked directly to the basis functions of a nonlinear regression expansion. All the used neural models allow a form of clustering for nonlinear basis function expansion, this is particularly true for Radial Basis Function (RBF) and the proposed fuzzy neural network illustrated in Sect. III.

From a practical point of view, the problem to be solved is the prediction prior to time t of the sample y_t of the return series and the prediction of the related volatility. In the paper, these problems are being also refereed to, indifferently, as the estimation of the conditional mean μ_t and the conditional variance σ_t^2, respectively. The main data base for the financial analysis is the collection in the past of the price series S_k, $k = 0 \dots (t-1)$, where price S_0 is the first sample ever collected in the past. By using (1), the data base of returns y_k, $k = 1 \dots (t-1)$, is obtained. Usually, the analysis at any time t makes use of a training set consisting of a limited number N_T of previous observations. As explained in the following, the training set is determined by means of previous models and predictions, using the samples y_k, ϵ_k, σ_k^2, $k = (t-N_T) \dots (t-1)$.

A prediction process is usually applied for a given number N_S of time intervals starting at time $t = T_S$, i.e. for $t = T_S \dots (T_S + N_S - 1)$ and $1 \leq N_T \leq T_S - 1$, where N_S also represents the horizon of prediction. Two alternatives are possible in this regard: the prediction models are estimated only once, prior to time $t = T_S$; the prediction models are estimated at every time step, by changing consequently the training set at any time. We will consider in the following the second approach, since the former is a generalization of a multi-step-ahead prediction for which suited techniques have been proposed in the literature.

The data-driven regression for time series prediction can be summarized in the iteration of the following steps [10]:

- *Initialization*. Let $t = T_S$ and find the initial conditions for ϵ_k and σ_k^2, $k = (T_S - N_T) \dots (T_S - 1)$. These values can be inferred by using any adequate technique. We used in this regard an ARMA-GARCH model applied to the samples y_k from $T_S - N_T$ to $T_S - 1$.
- *Step 1*. With the current value of t, determine the training set to be used for the model learning. It consists of two matrices $\mathbf{D}_t^{(\mu)}$ and $\mathbf{D}_t^{(\sigma)}$, where:
  - $\mathbf{D}_t^{(\mu)}$ is a $N_T \times (R+M+1)$ matrix whose the ith row $\mathbf{d}_{t,i}^{(\mu)}$, $i = 1 \dots N_T$, is
  $$\mathbf{d}_{t,i}^{(\mu)} = \left[\mathbf{x}_{t-N_T+i-1}^{(\mu)} \quad y_{t-N_T+i-1} \right] ; \tag{7}$$
  - $\mathbf{D}_t^{(\sigma)}$ is a $N_T \times (P+Q+1)$ matrix whose the ith row $\mathbf{d}_{t,i}^{(\sigma)}$, $i = 1 \dots N_T$, is
  $$\mathbf{d}_{t,i}^{(\sigma)} = \left[\mathbf{x}_{t-N_T+i-1}^{(\sigma)} \quad \sigma_{t-N_T+i-1}^2 \right] . \tag{8}$$

  Each row of these matrices is an input-output pattern that can be used for learning. In fact, the first $M+N$ columns of $\mathbf{D}_t^{(\mu)}$ and the first $P+Q$ columns of $\mathbf{D}_t^{(\sigma)}$ represent the inputs to h_μ and h_σ, respectively, for every sample of the training set. The last column of both matrices is the expected value to be estimated in correspondence with every pattern. The last row of matrices holds the most recent observation.
- *Step 2*. Determine, at the current time t, the parameters $\boldsymbol{\omega}_t^{(\mu)}$ of the regression function h_μ by using the training matrix $\mathbf{D}_t^{(\mu)}$ and an appropriate learning algorithm according to the chosen regression model. Similarly, learn the parameters $\boldsymbol{\omega}_t^{(\sigma)}$ of h_σ by using $\mathbf{D}_t^{(\sigma)}$. For example, if an ARMA-GARCH model is used, the parameters can be estimated by maximum Gaussian likelihood [8]. The specific procedure for the proposed fuzzy neural network is illustrated in the next section.
- *Step 3*. By means of the parameters $\boldsymbol{\omega}_t^{(\mu)}$ and $\boldsymbol{\omega}_t^{(\sigma)}$ determined in the previous Step 2, apply (5) and (6) to forecast the conditional mean μ_t and the volatility σ_t^2, respectively. Then, let $\epsilon_t = y_t - \mu_t$, $t \leftarrow t+1$, and go back to Step 1 if $t \leq T_S$.

Once the iteration is stopped, we have N_S samples of conditional mean (forecast), innovation and volatility pertaining to the time interval where prediction is carried out. The performance of prediction can be evaluated by means of suited benchmarks and error measures applied to the obtained results. A useful collection of such measures will be illustrated in Sect. IV.

III. The Neural Network for Data Regression

We introduce in the following the architecture of the fuzzy neural network to be used for regression in (5) and (6).

It is based on a generalized Sugeno-type fuzzy inference system (FIS), which is suited to approximate any input-output function by using a novel approach to estimated the parameters of high order consequents of C different fuzzy rules. The kth rule, $k = 1 \dots C$, has the following form:

$$\begin{aligned} &\text{If } x_1 \text{ is } B_1^{(k)} \text{ and } \dots \text{ and } x_n \text{ is } B_n^{(k)} \text{ then} \\ &y^{(k)} = h\left(\mathbf{x}; \boldsymbol{\omega}^{(k)}\right), \end{aligned} \tag{9}$$

where $\mathbf{x} = [x_1\ x_2\ \dots\ x_n]$ is a row vector (or pattern) in the n-dimensional input space[1] and $y^{(k)}$ is the scalar output associated with the rule. The latter is characterized by the membership functions (MFs) $\mu_{B_j^{(k)}}(x_j)$ of the fuzzy input variables $B_j^{(k)}$, $j = 1 \dots n$, and the set of parameters $\boldsymbol{\omega}^{(k)}$ of the related crisp output function.

Several alternatives are possible for the fuzzification of crisp inputs, the composition of input MFs, and the way rule outputs are combined [11]. Usually, the structure of the fuzzy inference system is the following one:

$$\tilde{y} = \frac{\sum_{k=1}^{C} \mu_{\mathbf{B}^{(k)}}(\mathbf{x})\, y^{(k)}}{\sum_{k=1}^{C} \mu_{\mathbf{B}^{(k)}}(\mathbf{x})}, \tag{10}$$

where $\mathbf{B}^{(k)}$ is the overall fuzzy input variable, $\mu_{\mathbf{B}^{(k)}}(\mathbf{x})$ is the corresponding MF, and $\tilde{y}$ the output estimated for an input $\mathbf{x}$.

We propose in the following a learning procedure suited to data regression and financial time series forecasting, by using a generalization of a well-known fuzzy neural model falling within this framework, that is the Adaptive Neuro-Fuzzy Inference Systems (ANFIS), in which Sugeno first-order type (linear) output functions are adopted. With the proposed approach any kind of output function can be used, in particular we will focus on quadratic structures and we will prove that they are able to obtain significant improvements in spite of the increase of the model complexity. Consequently, the proposed model will be referred to in the following as 'Higher-Order Neuro-Fuzzy Inference System' (HONFIS); its parameters will be estimated though a data-driven approach and a constructive procedure is adopted in order to find a suitable number of fuzzy rules.

The numerical parameters of rules are obtained through a learning process and using a training set of N_T input-output pairs $(\mathbf{x}_t, y_t)$, $t = 1 \dots N_T$. The crucial problem during the learning is to obtain a good generalization capability. This issue has been deeply investigated in the literature; usually, the generalization capability is optimized by checking data set for overfitting model validation [12]. In this case, the generalization capability is maximized only if HONFIS consists of a suitable number of rules. However, the determination of the optimal number is a very critical problem to be solved, also considering the increased complexity of rule output functions, since the neural network might be easily overfitted in case of noisy or ill-conditioned data. In this paper, a constructive procedure for the automatic determination of rules is also proposed. It aims to a regularization of the network architecture based on learning theory and hyperplane clustering-based techniques.

[1]In the present case, either $n = R + M$ or $n = P + Q$, as for (5) or (6), respectively.

A. Synthesis of Rules by Clustering

When dealing with numerical data, the rules of HONFIS networks are commonly synthesized using clustering techniques that can reduce the redundancy of data so that significant rules are determined directly from the clusters modeling the training set. The main drawback due to conventional clustering approaches is that induced clusters do not always reflect the real data structure. An innovative approach, dubbed Hyperplane Clustering Synthesis (HCS), was firstly proposed in [13]. By HCS, clusters are determined in joint input-output space, with the rules' shape corresponding to the underlying function adopted for the output, that was a hyperplane in the case of ANFIS linear rules.

A similar approach is herein proposed, with the aim to generalize the HCS algorithm also in case of HONFIS networks with general output functions. The basic steps for clustering in the joint input-output space, which is fundamentally an alternating optimization technique aiming to identify the cluster prototypes, are summarized in the following. Let $\Gamma = \{\Gamma_1, \Gamma_2, \dots, \Gamma_C\}$ be a set of C clusters (each associated with a rule output) and let every pattern of the training set be assigned randomly to one of these clusters. Then, the clustering procedure with C prototypes is based on the following iterative steps:

- *Step 1*. The coefficients $\boldsymbol{\omega}^{(k)}$, $k = 1 \dots C$, of each rule consequent are evaluated by solving a set of (generally) nonlinear equations; the generic equation is:

$$y_t = h\left(\mathbf{x}_t; \boldsymbol{\omega}^{(k)}\right), \tag{11}$$

  where y_t is the output associated with the input $\mathbf{x}_t$, as for (7) or (8), respectively; index 't' spans only the pairs of the training set assigned to the kth cluster. In this regard, we propose to solve the set of nonlinear equations in (11) and estimate the coefficients for the nonlinear regression using an iterative least squares estimation [14].
- *Step 2*. The assignment of patterns to clusters is updated. Each pair $(\mathbf{x}_t, y_t)$, $t = 1 \dots N_T$, of the training set is now assigned to the cluster Γ_q, with q such that:

$$\begin{aligned} d_t &= \left| y_t - h\left(\mathbf{x}_t; \boldsymbol{\omega}^{(q)}\right) \right| = \\ &= \min_{k=1\dots C} \left| y_t - h\left(\mathbf{x}_t; \boldsymbol{\omega}^{(k)}\right) \right|. \end{aligned} \tag{12}$$

  Evidently, a pattern is assigned the cluster having the minimum distance from it.
- *Step 3*. For every cluster Γ_k the *local* approximation error is evaluated:

$$D_k = \frac{1}{N_k} \sum\nolimits_t d_t, \tag{13}$$

where index 't' spans only the N_k pairs of the training set assigned lately (i.e., in Step 2) to the kth cluster.

- *Step 4*. The convergence is based on the quantity:

$$\Theta = \frac{\left| D - D^{(old)} \right|}{D^{(old)}} , \tag{14}$$

where D is the *global* approximation error over the whole data set in the current iteration defined by:

$$D = \frac{1}{N_T} \sum_{t=1}^{N_T} d_i , \tag{15}$$

and $D^{(old)}$ is the global approximation error calculated in the previous iteration. If Θ is less than a predetermined threshold θ, the clustering algorithm is stopped. Otherwise, the iteration goes back to Step 1 by using the current updated association of patterns to clusters. The default value $\theta = 0.01$ will be used in the following.

The clustering algorithm described so far yields only the consequents of the Sugeno-type fuzzy rules. To achieve the complete structure of the HONFIS network, it is mandatory to determine the firing strengths of the rules' antecedents. To this end, once the iterations of clustering have stopped, each pattern of the training set can be labeled with an integer q, $1 \leq q \leq C$, representing the rule it has been assigned during Step 2 of the last iteration. By using the labeled training set, a classification problem can be solved in the input space only; at the end of the training process, the input space will be tiled by a classification model able to assign a fuzzy label $L(\underline{x})$ to any pattern $\mathbf{x}$ of the input space:

$$L(\mathbf{x}) = [\mu_{\mathbf{B}^{(1)}}(\mathbf{x}) \;\; \mu_{\mathbf{B}^{(2)}}(\mathbf{x}) \;\; \ldots \;\; \mu_{\mathbf{B}^{(M)}}(\mathbf{x})] , \tag{16}$$

where the kth element of $L(\mathbf{x})$ represents the fuzzy membership of the pattern to the kth class and hence, it can be assumed as the firing strength $\mu_{\mathbf{B}^{(k)}}(\mathbf{x})$ of the kth rule associated with the hyperplane corresponding to that class.

When the output $\tilde{y}$ must be estimated for any input $\mathbf{x}$ during the normal HONFIS operation (i.e., testing), the classifier is used to determine the fuzzy label (16) using only the input value $\mathbf{x}$; then, the output $\tilde{y}$ is calculated using (10) by means of the firing strengths contained in the fuzzy label $L(\mathbf{x})$ and the output consequents in (9), whose parameters have been early determined by clustering in the joint input-output space. In this phase it is not necessary, in general, to assign the test sample $\mathbf{x}$ a specific hyperplane cluster. In the following, we will adopt a K-nearest neighbor (K-NN) strategy for classification within HONFIS networks. Namely, let $\mathbf{x}_{t_1}, \mathbf{x}_{t_2}, \ldots, \mathbf{x}_{t_K}$ be the K patterns of training set that score the smallest Euclidean distance from $\mathbf{x}$; then, the fuzzy label of $\mathbf{x}$ will be determined as:

$$L(\mathbf{x}) = \frac{1}{K} \sum_{q=1}^{K} L(\mathbf{x}_{t_q}) . \tag{17}$$

In the following the default value $K = 3$ will be used for K-NN classification into the input space.

The use of clustering in HONFIS networks is suited to capture volatility clustering in energy commodity prices. GARCH models are able to model volatility clustering mainly because the model parameters are estimated repeatedly over time. This is also obtained by our proposed approach. However, the standard GARCH model in (3) is only a linear one, which unlikely can capture the clusters present inside the training set. Consequently, this makes very critical the choice of the number of past observations to be used for prediction. Nonlinear GARCH models can alleviate this problem, although using a global nonlinear regression. The training algorithm of HONFIS networks is intended to find such clusters as its primary goal; nonetheless, the capability of HONFIS to find volatility clustering and nonlinear phenomena in the energy commodity time series will be clearly proved in Sect. IV.

B. Optimization of the HONFIS Structure

When training a HONFIS network, the main problems are the local convergence of estimation algorithms and the correct determination of the number C of rules. The former problem mainly depends on a good (usually random) initialization of numerical parameters associated with each rule[2]. The latter one is a well-known problem, which is directly related to the generalization capability of the neurofuzzy network and it can also be referred to as 'structural optimization' problem. In fact, the HONFIS performance could be inadequate if the training set is either underfitted or overfitted by a lacking or an excessive number of rules, respectively.

A plain solution to these problems is based in this paper on the use of the clustering algorithm with different values of C and with different initializations for every value of C. Once the set of different HONFIS networks is generated, the best network can be chosen by relying on the supervised nature of the learning procedure, i.e. by using a cost function measuring the overall generalization capability of the network in terms of its complexity and its approximation error. Basic concepts of learning theory can be adopted in this regard [11], [12]. Namely, the ANFIS network achieving the best generalization capability is the one that, under the same performance on the training set, is characterized by the lowest number of rules.

As a measure of the network performance on the training set, the mean squared error (MSE) is adopted:

$$E = \frac{1}{N_T} \sum_{t=1}^{N_T} (y_t - \tilde{y}_t)^2 , \tag{18}$$

where $\tilde{y}_t$ is the output generated by the HONFIS network in correspondence to the tth input pattern $\mathbf{x}_t$ of the training set. The optimal network is selected by using the following cost function depending upon the number of HONFIS rules:

$$F(C) = (1 - \lambda) \frac{E(C) - E_{\min}}{E_{\max} - E_{\min}} + \lambda \frac{C}{N_T} , \tag{19}$$

where $E_{\min}$ and $E_{\max}$ are the extreme values of the performance E that are encountered during the analysis of the

[2]In the present case, the initialization is carried out by assigning the patterns to clusters randomly before the clustering iteration starts.

different HONFIS networks; λ is a weight $0 \leq \lambda \leq 1$. This weight is not critical, since the results are slightly affected by its variation in a large interval centered in 0.5. Evidently, for a given value $C = \bar{C}$, the HONFIS network showing the best value $F(\bar{C})$ in (19) will be the one whose initialization and the successive clustering iterations yield the best performance $E(\bar{C})$ on the training set.

The plain maximization of (19) is obtained by a constructive procedure, where the number of rules is increased from 1 to $C_{\max}$, being $C_{\max}$ a fraction of the training set cardinality N_T that represents the maximum complexity allowed to the network. For each value of C, different initializations (i.e., starting values of the network parameters) are considered and several HONFIS networks are generated through the previous clustering procedure. In fact, the latter is based on a random initialization and hence, different initializations might produce different networks for the same value of C because of the presence of local minima. If different T initializations are carried out for each value of C, the optimization procedure will generate $TC_{\max}$ networks and the optimal one will be selected according to (19).

IV. Performance Evaluation

The validity of the proposed approach has been validated by means of extensive computer simulations. Some illustrative examples are summarized in this section, where actual return series related to energy commodity prices are used. The numerical results are obtained by using well-known neural and fuzzy neural models, which are compared with respect to the commonly used combination of ARMA-GARCH models estimated by maximum Gaussian likelihood.

The training procedure of neural regression models follows the scheme illustrated in Sect. II. Bearing in mind the introduced notation, let y_t, $t = T_S \dots (T_S + N_S - 1)$, be the set of actual samples of the time series to be predicted. For each sample, the neural networks are trained using the previous N_T observations of the time series, i.e. from $t - N_T$ to $t-1$, together with the related innovations and conditional variances. A one-step-ahead prediction is therefore applied and this procedure is iterated N_S times. After every prediction, the sequences of innovations, conditional mean and conditional variance forecast are updated. As mentioned, we adopt quadratic functions within HONFIS networks for the output of the kth rule, $k = 1 \dots C$:

$$\begin{aligned} h\left(\mathbf{x}; \boldsymbol{\omega}^{(k)}\right) = & \omega_{21}^{(k)} x_1^2 + \dots + \omega_{2n}^{(k)} x_n^2 + \\ & \omega_{11}^{(k)} x_1 + \dots + \omega_{1n}^{(k)} x_n + \omega_0^{(k)} . \end{aligned} \tag{20}$$

In addition to the proposed HONFIS neural network, structurally optimized by the constructive procedure previously illustrated, we further consider three well-known neural architectures: RBF, MoG and ANFIS. The former is a feed-forward neural network trained by a constructive procedure that iteratively creates a radial basis network one neuron at a time. Neurons are added to the network until an error goal or a maximum number of neurons is reached [15]. The Mixture of Gaussian (MoG) neural network is particularly suited for ill-posed and non-convex approximation and prediction tasks; it is trained by an Expectation-Maximization (EM) algorithm [16]. The ANFIS network, which use linear consequent functions in (9), is trained by a subtractive clustering method for rule extraction [17], while the rule parameters are obtained by means of a standard least-squares method coupled with the back-propagation optimization [11]. All the training procedures also aim to optimize the structural complexity (i.e., number of kernels, hidden nodes, fuzzy rules, etc.) of the resulting neural network by cross-validation. For reasons of conciseness we will not provide details about complexity in the following, since it is optimal as long as the neural network exhibits a good generalization capability, which is evaluated as described in the following by means of the network's performance on test sets not used during training.

We take particular care to the criteria adopted to evaluate the performance of the algorithms. Let μ_t be the set of conditional means representing the prediction (obtained by using any model) of the corresponding values y_t, $t = T_S \dots (T_S + N_S - 1)$. The error measures used in this paper are the following ones:

- Mean Squared Error (MSE):

$$\mathrm{MSE} = \frac{1}{N_S} \sum\nolimits_t (y_t - \mu_t)^2 ; \tag{21}$$

- Normalized Mean Squared Error (NMSE):

$$\mathrm{NMSE} = \frac{\sum_t (y_t - \mu_t)^2}{\sum_t (y_t - \bar{y})^2} ; \tag{22}$$

- Noise-to-Signal Ratio (NSR):

$$\mathrm{NSR}_{\mathrm{dB}} = 10 \log_{10} \frac{\sum_t (y_t - \mu_t)^2}{\sum_t y_t^2} ; \tag{23}$$

where $\bar{y}$ is the average of the actual N_S samples of y_t. Evidently, the lower is the value of all these errors the better is the prediction accuracy.

A successful model describing spot prices dynamics must capture the statistical features of the analyzed data in the simulated series. To this aim, the unconditional moments from the first up to the fourth order are estimated (as time averages) and considered for both actual and predicted sequences. A given model is suited to forecast and model energy commodity prices when the first four moments of the predicted sequences result as close as possible to the moments estimated on the market data. Being able to reproduce the probability distribution of the observed series together with an accurate prediction of the daily prices will allow investors and risk managers to estimate the profit/loss scenarios to set up the adequate risk management strategies.

We consider the time series obtained from the daily prices of specific energy commodities. We carried out several tests to assess the validity of the proposed approach. Coal, natural gas, crude oil, and electricity prices over the period 2001-2010 were collected for both the European and the US markets. The results reported in this paper refer to the US market only, similar results have been obtained for the

TABLE I

NUMERICAL RESULTS FOR DJUSCL RETURN SERIES

Model	Errors			Unconditional Moments			
	MSE	NMSE	NSR	Mean	Variance	Skewness	Kurtosis
DJUSCL				−0.31	0.74	−0.41	5.60
GARCH	0.80	1.08	0.33	0.34	0.33	0.07	9.36
RBF	0.76	1.03	0.13	−0.39	0.88	−0.18	5.28
MoG	0.78	1.04	0.18	0.77	1.01	−0.32	5.20
ANFIS	0.77	1.04	0.17	−0.27	0.91	−0.41	5.34
HONFIS	0.76	1.04	0.17	−0.29	0.81	−0.35	5.52

MSE, mean, and variance are scaled by 10^{-3}

European data set not showing significant differences in the forecasting ability. The studied commodities and the related indexes are: coal (DJUSCL, in \$/ton); Henry Hub natural gas (HH, in \$/MMBtu); crude oil (WTI, in \$/barrel), and electricity (PJM, in \$/MWh). For electricity prices we chose the peak load contract referred to h. 12:00. For each commodity price log-returns are estimated using (1). We chose a well representative time window across the 'very critical' year 2008, i.e. from the beginning of 2006 to the end of 2009. So, taking into account that we have approximately 250 prices and 249 returns per year in the case of coal, natural gas and crude oil series, each return series consists of about 1000 samples. In the case of electricity prices we have a series of 365 data, given that electricity is traded every day of the year; in this application, for comparison purposes, we adjust the series to 250 trading days only. Each model is trained on the previous $N_T = 500$ samples (almost two years) and $N_S = 500$ samples are predicted, i.e. the last two years starting from $t = 501$ to the last sample of 2009.

The prediction errors of the conditional mean are evaluated; in addition, the four unconditional moments are estimated for both the predicted sequences and the related original series. Prior investigations can be made in order to find the best combination of the model orders and the size of the training set as well; a fine tuning for the optimal estimate of every model can be addressed in future research works. In order to obtain an accurate comparison of the performances obtained by the proposed neural networks with respect to standard models, we carried out a preliminary optimization of the main model parameters, i.e. R, M, P, and Q, so as to obtain the best performance of the reference 'GARCH' model for a given time series. Then, every model will use the same parameters when applied to the same time series.

The optimal parameters of the coal DJUSCL returns are $R = 1$, $M = 1$, $P = 5$, $Q = 1$; hence, a GARCH reference model ARMA$(1, 5)$-GARCH$(1, 1)$ is fitted. The numerical results are summarized in Table I: all the neural models score a prediction error better than GARCH; RBF obtains the best NSR but the skewness is not properly matched as in the case of ANFIS and HONFIS that have a comparable performance. HONFIS achieves a good NSR performance of 0.17 dB and it is able to follow the dynamics of both conditional mean and increasing volatility, as proved by the behavior of the estimated conditional variance shown in Fig. 1.

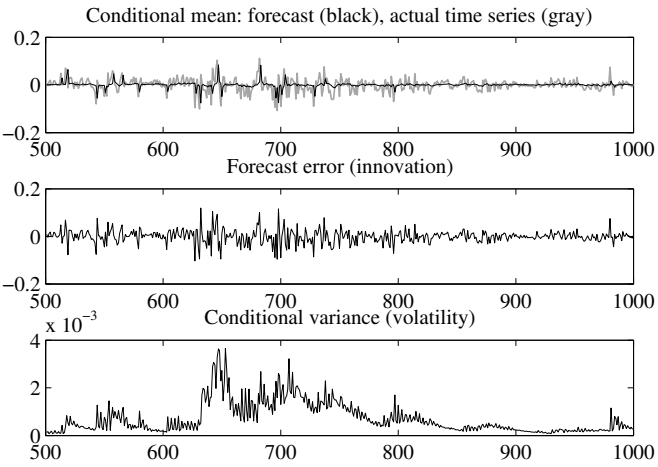

Fig. 1. Prediction of coal returns using the HONFIS neural network.

TABLE II

NUMERICAL RESULTS FOR HH RETURN SERIES

Model	Errors			Unconditional Moments			
	MSE	NMSE	NSR	Mean	Variance	Skewness	Kurtosis
HH				−0.46	1.72	1.46	10.62
GARCH	1.97	1.14	0.57	1.65	0.12	2.49	67.15
RBF	1.87	1.09	0.37	−0.44	1.92	0.27	4.54
MoG	1.80	1.05	0.21	−0.34	1.90	1.13	9.94
ANFIS	2.05	1.19	0.75	−0.47	1.87	0.71	7.35
HONFIS	1.67	1.03	0.13	−0.45	1.77	0.99	8.46

MSE, mean, and variance are scaled by 10^{-3}

The numerical results for HH returns of natural gas are reported in Table II. The optimal parameters are in this case $R = 2$, $M = 2$, $P = 2$, $Q = 1$, so a GARCH reference model ARMA$(2, 2)$-GARCH$(2, 1)$ is fitted. The HONFIS neural network has the best NSR performance of 0.13 dB and the related moments adequately fit with those of the original time series. A sufficient accuracy is also obtained by RBF and ANFIS neural networks. The GARCH is not suitable for the prediction of HH returns, since the moments are estimated very poorly, especially the kurtosis. The numerical results of the HONFIS neural network are qualitatively confirmed by the accurate predictions reported in the plots of Fig. 2, especially in the case of volatility.

The large volatility of crude oil WTI returns at the end of 2008 is the feature that requires an accurate forecasting technique. A more complex model is therefore necessary, using $R = 4$, $M = 2$, $P = 2$, $Q = 3$. The GARCH reference model ARMA$(4, 2)$-GARCH$(2, 3)$ is evidently outperformed by the neural networks, as evidenced by the results summarized in Table III. The best NSR is once again obtained by HONFIS, although the predicted sequence does not fit the skewness of the original one; the ANFIS performance suffers from the same drawback. The MoG network is able to fit the original moments, also maintaining a good prediction accuracy and following the changes of volatility in the underlying process.

Finally, the numerical results for the returns of PJM elec-

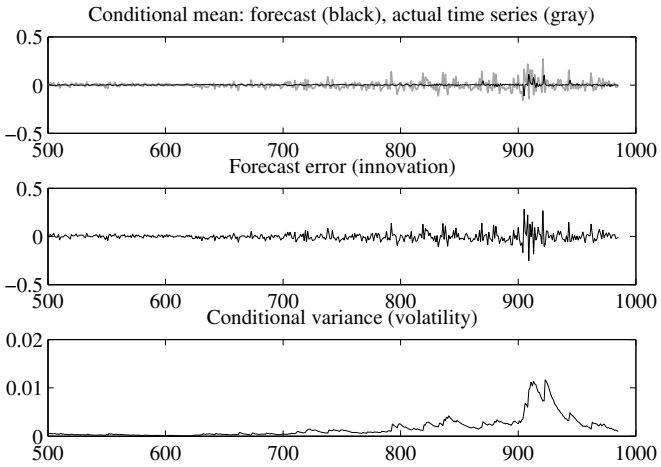

Fig. 2. Prediction of natural gas returns using the HONFIS neural network.

TABLE III
NUMERICAL RESULTS FOR WTI RETURN SERIES

Model	Errors			Unconditional Moments			
	MSE	NMSE	NSR	Mean	Variance	Skewness	Kurtosis
WTI				−0.10	0.11	0.34	6.51
GARCH	0.12	1.00	0.01	0.43	0.01	0.48	9.38
RBF	0.11	0.94	−0.27	−0.01	0.12	0.16	4.06
MoG	0.11	0.92	−0.36	−0.10	0.10	0.44	5.73
ANFIS	0.09	0.89	−0.50	−0.05	0.12	−0.25	7.22
HONFIS	0.08	0.85	−0.71	−0.08	0.13	−0.01	6.98

MSE, mean, and variance are scaled by 10^{-3}

TABLE IV
NUMERICAL RESULTS FOR PJM RETURN SERIES

Model	Errors			Unconditional Moments			
	MSE	NMSE	NSR	Mean	Variance	Skewness	Kurtosis
PJM				0.22	1.21	0.38	5.52
GARCH	1.19	0.98	−0.09	1.53	0.24	0.53	8.10
RBF	0.71	0.58	−2.36	−0.11	1.64	−0.01	2.96
MoG	0.57	0.47	−3.28	−0.41	1.77	0.28	4.61
ANFIS	0.64	0.53	−2.76	−0.21	1.24	0.26	5.58
HONFIS	0.60	0.51	−2.92	−0.17	1.21	0.29	5.46

MSE, mean, and variance are scaled by 10^{-3}

tricity index are reported in Table IV. The model parameters are $R = 3$, $M = 2$, $P = 1$, $Q = 1$; the GARCH reference model is ARMA$(3, 2)$-GARCH$(1, 1)$. The MoG neural network performs better than the other models in this case. Anyway, the proposed HONFIS model performs better than the original ANFIS network. Globally, neural networks improve the NSR performance of more than 2 dB with respect to GARCH, despite a biasing that shifts the estimate of the mean to negative values.

V. CONCLUSION

A new neural network approach is proposed for modeling time series associated with energy commodity prices, which is based on fuzzy neural networks using higher-order Sugeno-type consequent rules. The use of a constructive procedure determining automatically the optimal number of fuzzy rules is also illustrated in order to avoid overfitting and maximize the generalization capability of the neural network.

The proposed approach provides an accurate description of energy prices dynamics, allowing us to estimate daily prices for energy commodities over a long time horizon. The validation performed on historical data shows that the neural network approach generates prices that are able to replicate the daily data and to reproduce the same probability distribution for the various series. The proposed HONFIS model, using quadratic consequent rules, outperforms the original ANFIS network in almost all cases, making useful significantly the increased complexity of the related model.

Currently, we are investigating more advanced techniques for the application of the proposed approach to a multivariate time series analysis and for the automatic and more reliable selection of the samples to be used for prediction, including the order of regression models and the resulting complexity of neural models.

REFERENCES

[1] J. Hamilton, "Understanding crude oil prices," *Energy Journal*, vol. 20, no. 2, pp. 179–206, 2009.

[2] R. Gibson and E. Schwartz, "Stochastic convenience yield and the pricing of oil contingent claims," *The Journal of Finance*, vol. 45, pp. 959–976, 1990.

[3] C. Mari, "Random movements of power prices in competitive markets: a hybrid model approach," *The Journal of Energy Markets*, vol. 1, no. 2, pp. 87–103, 2008.

[4] T. Mills, *Time Series Techniques for Economists*. Cambridge, UK: Cambridge University Press, 1990.

[5] I. Haidar, S. Kulkarni, and H. Pan, "Forecasting model for crude oil prices based on artificial neural networks," in *Proc. of International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP 2008)*, Sydney, NSW, Australia, 2008, pp. 103–108.

[6] M. Panella, F. Barcellona, and R.L. D'Ecclesia, "Subband prediction of energy commodity prices," in *Proc. of the IEEE Int. Workshop on Signal Processing Advances in Wireless Communications (SPAWC 2012)*. Cesme, Turkey: IEEE, 2012.

[7] M. Panella, L. Liparulo, F. Barcellona, and R.L. D'Ecclesia, "A study on crude oil prices modeled by neurofuzzy networks," in *Proc. of FUZZ-IEEE 2013*, Hyderabad, India, 2013.

[8] T. Bollerslev, "Generalized autoregressive conditional heteroskedasticity," *Journal of Econometrics*, vol. 31, p. 307327, 1986.

[9] R. Engle, "Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation," *Econometrica*, vol. 50, no. 4, pp. 987–1007, 1982.

[10] M. Panella, F. Barcellona, and R.L. D'Ecclesia, "Forecasting energy commodity prices using neural networks," *Advances in Decision Sciences*, vol. 2012, 2012.

[11] J.-S. Jang, C. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing: a Computational Approach to Learning and Machine Intelligence*. Upper Saddle River, NJ, USA: Prentice Hall, 1997.

[12] S. Haykin, *Neural Networks, a Comprehensive Foundation, 2nd Edition*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1999.

[13] M. Panella, "A hierarchical procedure for the synthesis of ANFIS networks," *Advances in Fuzzy Systems*, vol. 2012, 2012.

[14] G. Seber and C. Wild, *Nonlinear Regression*. NJ: Wiley-Interscience: Hoboken, 2003.

[15] A. Bors and I. Pitas, "Median radial basis function neural network," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1351–1364, 1996.

[16] M. Panella, "Advances in biological time series prediction by neural networks," *Biomedical Signal Processing and Control*, vol. 6, no. 2, pp. 112–120, 2011.

[17] S. Chiu, "Fuzzy model identification based on cluster estimation," *Journal of Intelligent & Fuzzy Systems*, vol. 2, pp. 267–278, 1994.