Long-Term Wind Speed and Power Forecasting Using Local Recurrent Neural Network Models

Thanasis G. Barbounis, John B. Theocharis, *Member, IEEE*, Minas C. Alexiadis, and Petros S. Dokopoulos, *Member, IEEE*

***Abstract*—This paper deals with the problem of long-term wind speed and power forecasting based on meteorological information. Hourly forecasts up to 72-h ahead are produced for a wind park on the Greek island of Crete. As inputs our models use the numerical forecasts of wind speed and direction provided by atmospheric modeling system SKIRON for four nearby positions up to 30 km away from the wind turbine cluster. Three types of local recurrent neural networks are employed as forecasting models, namely, the infinite impulse response multilayer perceptron (IIR-MLP), the local activation feedback multilayer network (LAF-MLN), and the diagonal recurrent neural network (RNN). These networks contain internal feedback paths, with the neuron connections implemented by means of IIR synaptic filters. Two novel and optimal on-line learning schemes are suggested for the update of the recurrent network's weights based on the recursive prediction error algorithm. The methods assure continuous stability of the network during the learning phase and exhibit improved performance compared to the conventional dynamic back propagation. Extensive experimentation is carried out where the three recurrent networks are additionally compared to two static models, a finite-impulse response NN (FIR-NN) and a conventional static-MLP network. Simulation results demonstrate that the recurrent models, trained by the suggested methods, outperform the static ones while they exhibit significant improvement over the persistent method.**

***Index Terms*—Local recurrent neural networks, long-term wind power forecasting, nonlinear recursive least square learning, real time learning.**

I. Introduction

WIND ENERGY conversion systems (WECS) appear as an appealing alternative to conventional power generation, being most appropriate for isolated power systems on islands or rural areas. Integration of accurate wind forecasts in the management routines involved in WECS provides a significant tool for optimizing operating costs and improving reliability.

However, due to highly complex interactions and the contribution of various meteorological parameters, wind forecasting is a severe task. Despite the difficulties, a variety of approaches have been suggested in the literature. The particular prediction method used depends on the available information and the time-scale of the application. Wind forecasts in the range of a few seconds are used for wind turbines control [1], [2]. Time scales in the order of several minutes or even hours are encountered, when power system scheduling is to be addressed [3]. In these cases, the time series approach is usually followed, as in [4] where a recurrent high-order neural network (NN) is used for short-term prediction of wind power. Furthermore, similar models have been applied for daily, weekly, or even monthly time series [5].

Long-term prediction of wind power allows planning the connection or disconnection of wind turbines or conventional generators, thus achieving low spinning reserve and optimal operating cost. It refers to hourly data and a time horizon of up to two to three days ahead. In such cases, the statistical properties of the wind are not helpful, and hence, we have to rely on approximate wind forecasts provided by the national meteorological services. These predictions are calculated for some predefined reference points, not necessarily on the position of the park, so the need arises for the reduction of these predictions to the site of interest.

In the past, considerable efforts have been devoted to utilizing meteorological information to derive the required forecasts.

Micro- and Meso-scale models (such as WASP or PARK) are actually deterministic models that suggest various correcting parameters according to the terrain properties (orography, roughness). They also take consideration of the number, type, and location of wind turbines in the wind farm, the position, the hub-height, the power curve of each one to produce the total power output of the wind farm [6], [7].

Model output statistics (MOS) are used to translate numerical meteorological inputs to actual wind power outputs, e.g., simple methods to correct bias and scaling errors of the initial forecasted values [8]–[11]. Artificial Intelligence models are actually advanced MOS techniques combining complex input-output architecture, robust, and flexible adaptive algorithms [12].

In this paper, we deal with long-term wind speed and power forecasting for a wind park. Owing to the large time-scale, we are based on three-days ahead meteorological predictions (wind speed and direction) provided at four nearby sites of the park. To account for the complex dynamics of the process, three local recurrent neural networks with internal feedback paths are employed to produce 72–hours-ahead wind forecasts. The above choice is motivated by the fact that these models exhibit faster learning compared to the fully recurrent neural networks [13]. First, the infinite impulse response multilayer perceptron (IIR-MLP) is suggested, having sufficiently rich network architecture. Additionally, more relaxed structures are considered, including the local activation feedback multilayer

Manuscript received December 9, 2003; revised June 11, 2004. Paper no. TEC-00357-2003.

T. G. Barbounis and J. B. Theocharis are with the Electronic and Computer Engineering Division, Electrical and Computer Engineering Department, Aristotle University of Thessaloniki, Thessaloniki, Greece.

M. C. Alexiadis and P. S. Dokopoulos are with the Electrical Power Systems Laboratory, Electrical and Computer Engineering Department, Aristotle University of Thessaloniki, Thessaloniki, Greece.

Digital Object Identifier 10.1109/TEC.2005.847954

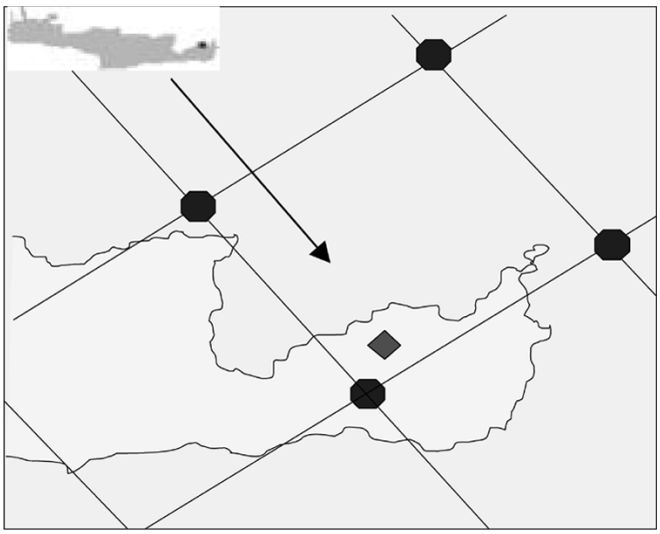

Fig. 1. Geographical location of the wind park of Rokas, on the Greek island of Crete. Also shown are the nodes where meteorological predictions are available and the prevailing wind direction.

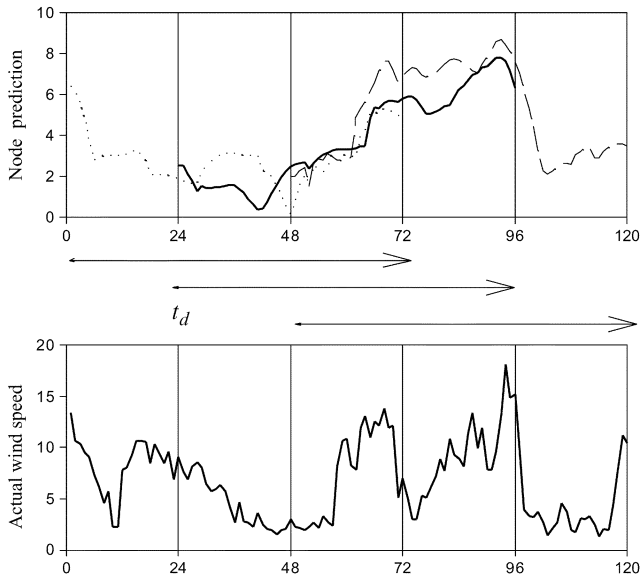

Fig. 2. Configuration of the forecasting approach. The node predictions and the wind forecasts produced by the models are given at the beginning of each day. They cover a time period of 72-h ahead and are updated every 24 h.

network (LAF-MLN) and the diagonal recurrent neural network (DRNN) model. Two novel and efficient learning schemes are developed, a global and a decoupled approach of the recursive prediction error (RPE) algorithm, for updating the network's weights, suitable for on-line applications. The experimental results show that accurate forecasts are obtained, with the recurrent forecast models exhibiting superior performance with regard to the static networks.

II. Problem Formulation

Let us consider the Rokas W/F with a capacity of 10.2 MW located at Eastern Crete, Greece. For efficient maintenance and resource planning of WECS, it is very helpful for us to know in advance the wind speed and power at the park, for a time horizon including a few days ahead. This allows an optimal policy to be designed, using the optimum amount of the available wind turbines and scheduling the possible need for storing or supplementing the generated power.

Due to the large time scale, the main source of information is the wind meteorological predictions, near-surface "node predictions", calculated for 4 specific positions (N, S, E, W) located around the wind park (see Fig. 1). The node predictions are given once per day, and assume for simplicity that they are available at the beginning of each day $d = 1, 2, \ldots, N$, where N is the number of days considered in the data set. For each node and day, the meteorological data are formulated as records comprising predictions of the wind speed and direction at the succeeding 72-h ahead

$$\hat{W}_d^{\text{node}} = \left\{ \left(\hat{w}_{\text{sp}}^{\text{node}}[t_d + h], \hat{w}_{\text{dir}}^{\text{node}}[t_d + h] \right) \middle| \, h = 1, \ldots, 72 \right\} \tag{1}$$

where $d = 1, 2, \ldots, N$, node $\in \{N, S, E, W\}$, and $\hat{w}_{\text{sp}}^{\text{node}}, \hat{w}_{\text{dir}}^{\text{node}}$ denote the meteorological node predictions of wind speed and direction, respectively, at time $t = t_d + h$. Also available are the predictions of atmospheric pressure and temperature at the park site.

The real data provided by the W/F SCADA system include values of the wind speed measured at a reference point within the park and the total power of the farm, denoted as $w_{\text{sp}}^r[t]$ and $w_{\text{pw}}^r[t]$, respectively. The data are recorded hourly from April 1st, 2000 until December 31st, 2000.

Given the node predictions, the objective is to develop an advanced forecast model providing 72-h-ahead wind speed and power forecasts at the park, denoted as $\hat{w}_{\text{sp}}$ and $\hat{w}_{\text{pw}}$, respectively. The configuration of the forecasting approach is depicted in Fig. 2. The wind estimates are given at the beginning of each day and the model is generally described by

$$\hat{y}(t) = \hat{y}(t_d + h) = F\{\hat{X}^{\text{node}}(t), \hat{Y}(t), \Theta\}, \quad h = 1, \ldots, 72 \tag{2}$$

where $\hat{y}(t)$ stands for either $\hat{w}_{\text{sp}}$ or $\hat{w}_{\text{pw}}$, $F\{\cdot\}$ represents the nonlinear mapping function of the process, and the vectors $\hat{X}^{\text{node}}(t)$ and $\hat{Y}(t)$ are given as

$$\begin{aligned} \hat{X}^{\text{node}}(t) = \{ & \hat{w}_{\text{sp}}^{\text{node}}(t), \hat{w}_{\text{sp}}^{\text{node}}(t-1), \ldots, \\ & \hat{w}_{\text{dir}}^{\text{node}}(t), \hat{w}_{\text{dir}}^{\text{node}}(t-1), \ldots \} \\ \hat{Y}(t) = \{ & \hat{y}(t-1), \hat{y}(t-2), \ldots \}. \end{aligned} \tag{3}$$

Apparently, there are both spatial and temporal correlations involved between the node predictions and the wind variables at the park to be forecasted, rendering the system a highly complex, dynamic, and nonstationary process. The wind values are affected by large-scale atmospheric conditions and the morphology of the surface landscape.

Efficient forecasting dictates that the model should exhibit the following properties. First, for each t, the current and past values of the node predictions should be considered as model inputs, as suggested by $\hat{X}^{\text{node}}(t)$, so that the model can properly identify the input trends and variations. Moreover, in the absence of real wind speed and power values for times greater than t_d, the model's previous estimates, $\hat{Y}(t)$, should be used

to derive the current output estimates, $\hat{y}(t)$. Finally, the model should be capable of memorizing the dynamic nature of the process. In this paper, we employ three types of recurrent neural networks, as advanced forecast models, to generate long-term estimates of $\hat{w}_{\rm sp}(t)$ and $\hat{w}_{\rm pw}(t)$. These networks belong to the family of local-recurrent global-feedforward (LRGF) models with internal dynamics, have strong temporal modeling capabilities and fulfill completely the quality criteria described above. Furthermore, two novel and efficient learning schemes are also devised for the update of the network weights.

When an advanced model is not available, the so-called persistent forecasts can be obtained with a minimal effort using the most recent information available. Following this approach, the forecast $\hat{y}^{\rm pers}(t_d+h\,|\,t_d)$ at h future time-steps is determined as an average of the h past values

$$\hat{y}^{\rm pers}(t_d+h\,|\,t_d)=\frac{1}{h}\sum_{j=0}^{h-1}y(t_d-j),\quad h=1,\ldots,72 \tag{4}$$

where $y(t_d-j)$ are real values of wind speed and power, measured at times prior to t_d. This naive predictor suggests that as the forecasting time lag increases, correlation with recent past measurements becomes negligible, so a longer scale average should be preferred instead. The advantage gained by an advanced model is referred to as % error improvement over persistent and serves as a means to evaluate the model's performance.

III. Network Architecture

The LRGF recurrent networks are composed of layers arranged in a feedforward fashion. Each layer contains dynamic processing units (neurons) with time-delay lines and/or feedback. Depending on the neuron dynamic model, three network types are mainly considered, namely, the IIR-MLP, the local activation feedback MLN (LAF-MLN) and the output feedback MLN. In this paper, we focus on the IIR-MLP and the LAF-MLN. Additionally, a simplification of the LAF-MLN is considered, namely, the DRNN.

A. The IIR-MLP Model

The IIR-MLP architecture, suggested by Tsoi and Back [13], consists of neurons where the synaptic weights are replaced with infinite impulse response (IIR) linear filters, also refereed to as autoregressive moving average (ARMA) models, as shown in Fig. 3(a).

To cope with the structural complexity of the networks, the following notational convention is employed. The IIR-MLP network is assumed to consist of $\ell=0,\ldots,\mathrm{M}$ layers, with $\ell=0$ and $\ell=\mathrm{M}$ denoting the input and the output layer, respectively. The ℓth layer contains N_ℓ neurons, N_0 and N_M being the number of neurons in the input and the output layer, respectively. $x_n^{(\ell)}[t]$ is the output of the nth neuron of the ℓth layer at time t. In particular, $x_n^{(0)}[t], n=1,\ldots,N_0$ are the network's input signals, while $x_n^{(M)}[t], n=1,\ldots,N_M$ are the output signals. $s_n^{(\ell)}[t]$ is the output of the summing point, that is, the input to the activation function of the nth neuron of the ℓth layer at time t. $y_{nm}^{(\ell)}[t]$ is the synaptic filter output at time t connecting the nth neuron in the ℓth layer with the mth neuron of the $(\ell-1)$th layer. $(L_{nm}^{(\ell)}-1)$, and $I_{nm}^{(\ell)}$ denote the order of the MA and the AR part, respectively, of the synapse connecting the nth neuron in the ℓth layer with the mth input coming from the $(\ell-1)$th layer, with $L_{nm}^{(\ell)}\geq 1, L_{n0}^{(\ell)}=1$, and $\mathrm{I}_{nm}^{(\ell)}\geq 0, \mathrm{I}_{n0}^{(\ell)}=0$. $w_{nm(p)}^{(\ell)}, p=0,\ldots,L_{nm}^{(\ell)}-1$ and $v_{nm(p)}^{(\ell)}, p=1,\ldots,I_{nm}^{(\ell)}$ are the coefficients of the MA and the AR, respectively, of the corresponding synapse. $w_{n0}^{(\ell)}$ is the bias term of each neuron. Finally, $\mathrm{sgm}(\cdot)$ and $\mathrm{sgm}'(\cdot)$ are the node's activation function and its derivative.

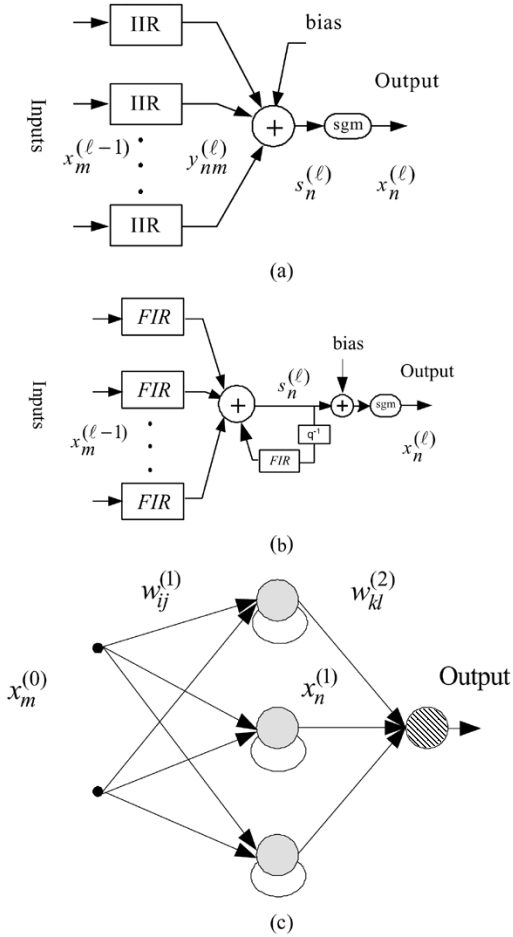

Fig. 3. Local recurrent NNs used for wind forecasting. (a) Neuron model for IIR-MLP. (b) Neuron model for LAF-MLN. (c) DRNN network configuration.

The forward run at time t, evaluated for $\ell=1,\ldots,M$ and $n=1,\ldots,N_\ell$, is described as follows:

$$y_{nm}^{(\ell)}[t]=\sum_{p=0}^{L_{nm}^{(\ell)}-1}w_{nm(p)}^{(\ell)}x_m^{(\ell-1)}[t-p]+\sum_{p=1}^{I_{nm}^{(\ell)}}v_{nm(p)}^{(\ell)}\,y_{nm}^{(\ell)}[t-p] \tag{5}$$

$$s_n^{(\ell)}[t]=\sum_{m=1}^{N_{\ell-1}}y_{nm}^{(\ell)}[t]+w_{n0}^{(\ell)} \tag{6}$$

$$x_n^{(\ell)}[t]=\mathrm{sgm}\left(s_n^{(\ell)}[t]\right) \tag{7}$$

Assuming that the network is running along a training sequence (epoch wise mode) with the weights remaining fixed throughout the epoch, the neuron's dynamics can be described

in a compact way using a notation employed in the adaptive filter theory, as follows:

$$y_{nm}^{(\ell)}[t] = \frac{B_{nm}^{(\ell)}(q^{-1})}{\left[1 - A_{nm}^{(\ell)}(q^{-1})\right]} x_m^{(\ell-1)}[t] \tag{8}$$

where

$$B_{nm}^{(\ell)}(q^{-1}) = \sum_{p=0}^{L_{nm}^{(\ell)}} w_{nm(p)}^{(\ell)} q^{-p} \tag{9}$$

$$A_{nm}^{(\ell)}(q^{-1}) = \sum_{p=1}^{I_{nm}^{(\ell)}} \upsilon_{nm(p)}^{(\ell)} q^{-p} \tag{10}$$

and q^{-1} is the delay operator, $q^{-p} y_{nm}^{(\ell)}[t] = y_{nm}^{(\ell)}[t-p]$. It can be seen that the IIR-MLP architecture realizes a sufficiently rich class of models. It includes several known neural types as special cases, depending on the parameter settings of the IIR synaptic filters. For instance, the IIR-LMP can be reduced to a FIR-NN [14] by eliminating the feedback connections ($\mathrm{I}_{nm}^{(\ell)} = 0$). Moreover, IIR-MLP reduces to the conventional static MLP [15] by discarding the MA and the AR parts of the synaptic filters ($L_{nm}^{(\ell)} = 1, \mathrm{I}_{nm}^{(\ell)} = 0$).

B. The LAF-MLN Model

The neuron model of the LAF-MLN structure [16] is shown in Fig. 3(b). The output of a neuron summing-node is filtered through an autoregressive (AR) adaptive filter before feeding the activation function. It should be noticed that, regarding the structural complexity, the LAF-MLN is considerably simpler compared to the IIR-MLP. This is due to the fact that while an AR part is introduced in every synaptic link in IIR-MLP, a single AR recursion exists for each neuron in the LAF-MLN architecture. As a result, for the same network structure, the latter network contains a smaller number of tunable weights compared to the former one.

The forward run equations for the LAF-MLN are described as follows:

$$s_n^{(\ell)}[t] = \sum_{m=1}^{N_{l-1}} \sum_{p=0}^{L_{nm}^{(\ell)}-1} w_{nm(p)}^{(\ell)} x_m^{(\ell-1)}[t-p] + \sum_{p=1}^{I_n^{(\ell)}} \upsilon_{n(p)}^{(\ell)} s_n^{(\ell)}[t-p] \tag{11}$$

$$x_n^{(\ell)}[t] = \mathrm{sgm}\left(\tilde{s}_n^{(\ell)}[t]\right) \tag{12}$$

$$\tilde{s}_n^{(\ell)}[t] = s_n^{(\ell)}[t] + w_{n0}^{(\ell)}. \tag{13}$$

C. The DRNN Model

In order to further reduce the structural complexity issue, we consider the DRNN, a modified form of the fully recurrent model [17]. DRNN is a two-layer network [Fig. 3(c)], where the hidden layer contains self-recurrent neurons while the output layer is composed of linear neurons. The hidden layer equations are

$$s_n^{(1)}[t] = \sum_{m=1}^{N_0} w_{nm}^{(1)} x_m^{(0)}[t-p] + \upsilon_n^{(1)} s_n^{(1)}[t-1] \tag{14}$$

$$x_n^{(1)}[t] = \mathrm{sgm}\left(s_n^{(1)}[t]\right) \tag{15}$$

while the network's output is determined by

$$x_q^{(2)}[t] = \sum_{m=1}^{N_1} w_{qn}^{(2)} x_n^{(1)}[t], \quad q = 1, \ldots, N_2. \tag{16}$$

Notice that the DRNN can be derived as a special case of the LAF-MLN for $M = 2$. Particularly, we consider only the constant terms of the MA filters ($L_{nm}^{(1)} = 1$) for all neurons. Moreover, the AR parts are reduced to first order filters ($I_n^{(1)} = 1$) for the hidden layer's neurons, while feedback is broken for the neurons in the output layer.

IV. Gradient Calculations

The learning algorithm to be developed in the next section requires the computation of the gradients of the network's output with respect to all trainable weights. Because of the existence of internal dynamics, the traditional procedure of the standard back propagation (BP) cannot be applied to determine these gradients. Therefore, we employ the method of ordered derivatives [18], extensively used in the literature for calculating partial derivatives in complex recurrent networks. Notice that since the recursive approach is adopted, the chain rule derivative expansion is developed in a forward fashion.

The gradients relate differential changes of the neuron outputs to differential changes of a weight within the network. Consider an arbitrary weight $u_{nm(p)}^{(\ell)}$ denoting either $w_{nm(p)}^{(\ell)}$ or $u_{nm(p)}^{(\ell)}$ of a synapse ending to the nth neuron of the ℓth layer. Derivation of the gradient calculations for the IIR-MLP model is given in the Appendix. The respective calculations for the LAF-MLN and the DRNN proceed along similar lines and therefore they are omitted.

Notice that the calculation of the gradients is achieved through higher order recurrent difference equations. This is opposed to static network structures where the weights are updated using static relations [15]. As regards the IIR-MLP model and based on the gradient analysis described in the Appendix, the following comments are in order. The gradients of the synaptic filter output $y_{nm}^{(\ell)}$ with regard to the filter weights $w_{nm(p)}^{(\ell)}$ and $u_{nm(p)}^{(\ell)}$ are derived by passing $x_m^{(\ell-1)}[t-p]$ and $y_{nm}^{(\ell)}[t-p]$, respectively, through the AR part of the synaptic filter. Furthermore, at each t, the gradients of the synaptic filter outputs $y_{qn}^{(\ell+1)}[t]$ belonging to the succeeding layer $(\ell+1)$ with respect to $\nu_{nm(p)}^{(\ell)}$ (w or u), where the weight change is assumed, are calculated by passing the corresponding gradients of the nth neuron's output at the ℓth layer through the filter IIR_{qn}. Hence, the gradient dynamics is identical to the forward one, describing the dynamics of $x_n^{(\ell)}[t]$ through the IIR_{qn} synapse to produce $y_{qn}^{(\ell+1)}[t]$. Finally, the gradients of the filter outputs $y_{uq}^{(\ell+i)}[t]$ of the $(\ell+i)$th layer with $i \geq 2$ are derived in terms of the gradients of the neuron outputs of the preceding layer $(\ell+i-1)$th with respect to $\nu_{nm(p)}^{(\ell)}$.

Following the above observations, the "gradient amidst dynamics" with respect to an arbitrary weight $\nu_{nm(p)}^{(\ell)}$ is described in terms of an auxiliary network, $S_{nm}^{(\ell)}$, called the sensitivity network [19]. For each $\nu_{nm(p)}^{(\ell)}$, the associated $S_{nm}^{(\ell)}$ is formed as a sub-network of the original one, starting with the n-th node of the ℓ-th layer and proceeding ahead until the output layer.

$S_{nm}^{(\ell)}$ is fed by the gradients of $x_{nm}^{(\ell)}$ with respect to $\nu_{nm(p)}^{(\ell)}$ while its output is the gradient of the network outputs with regard to $\nu_{nm(p)}^{(\ell)}$.

Notice that the gradient calculations are carried out, through the sensitivity networks, in parallel with the forward network, following similar dynamics. That is, the gradients are calculated on an on-line basis, as the original network runs in time. Hence, the use of the sensitivity networks provides a transparent tool for the evolution of the network gradients. It is assumed that during on-line training the weights are changing smoothly, that is, they remain almost constant. In that case, the weight gradients of higher order $p = 1, 2, \ldots$ can be obtained as delayed versions of the gradients with respect to $\nu_{nm(0)}^{(\ell)}$.

V. Learning Algorithm

Having determined the network gradients, we proceed to developing the learning algorithms for the on-line update of the tunable weights. The most common algorithm used for on-line training of recurrent networks is real-time recurrent learning (RTRL) [20], where the weight update is performed along the negative gradient direction. Nevertheless, this method exhibits slow convergence rates because of the small learning rates required, and most often becomes trapped to local minima. In an attempt to improve the performance, we employ an optimal learning scheme, the recursive prediction error (RPE) identification algorithm, with enhanced training qualities. Owing to the second order information used in the update recursions, better-conditioned training is accomplished compared to the RTRL method.

Along this direction, two novel algorithms are suggested for training the local feedback recurrent networks considered in this paper, where the stability issue is assured throughout the learning process. First a global scheme is developed, called the global RPE (GRPE), where all weights are simultaneously updated. Additionally, to cope with the increased computational complexity of the GRPE, we devised a local version of the algorithm, called the decoupled RPE (DRPE). The DRPE is derived by partitioning the global optimization problem into a set of manageable sub-problems, at the neuron level. Thus, considerable computational and storage savings are gained while preserving high accuracy qualities, similar to the ones of the GRPE.

A. The GRPE Algorithm

Let $\vartheta(t)$ denote an n-dimensional composite vector including all synaptic weights of the recurrent network under consideration. We consider a nonlinear recurrent predictor of the type

$$\hat{y}(t \,|\, \vartheta) = h(\vartheta, u(t), \varphi(t, \vartheta)) \tag{17}$$

where $h(\cdot)$ describes the structure of the network. $\hat{y}(t \,|\, \vartheta)$, a $(N_M \times 1)$ vector, comprises the network's outputs, $x_i^{(N_M)}, i = 1, \ldots, N_M$, $u(t)$ is a $(N_0 \times 1)$ vector including the network's inputs $x_i^{(N_0)}, i = 1, \ldots, N_0$, and $\varphi(t, \vartheta)$ represents the internal states $x_n^{(\ell)}[t]$ describing the network's dynamics. The real process to be modeled by the network, denoted as $y(t)$, is obtained by

$$y(t) = \hat{y}(t \,|\, \vartheta) + \varepsilon(t, \vartheta) \tag{18}$$

where $\varepsilon(t, \vartheta)$ is the prediction error for a particular value of ϑ. According to the GRPE method, all network weights are continuously determined at each t using the following recursion:

$$\varepsilon(t) = y(t) - \hat{y}(t \,|\, \hat{\vartheta}(t-1)) \tag{19}$$

$$S(t) = \psi(t)^T P(t-1)\psi(t) + \lambda I \tag{20}$$

$$L(t) = P(t-1)\psi(t)S^{-1}(t) \tag{21}$$

$$\hat{\vartheta}(t) = [\hat{\vartheta}(t-1) + \mu(t)L(t)\varepsilon(t)]_{D_m} \tag{22}$$

$$P(t) = [P(t-1) - L(t)S(t)L^T(t)]/\lambda \tag{23}$$

where λ is the forgetting factor, $0 < \lambda \leq 1$. The GRPE algorithm provides a recursive way of minimizing a quadratic criterion of the prediction errors using the stochastic Gauss-Newton search method [21]. The algorithm is identical to a nonlinear recursive least squares (RLS) [22] method that minimizes the error criterion

$$V_t(\vartheta) = \sum_{k=1}^{t} \lambda^{t-k} \{\varepsilon^T(k)\varepsilon(k)\}. \tag{24}$$

Furthermore, the recursion (19)–(23) has strong similarities to the extended Kalman filter (EKF) algorithm used in [23]. $L(t)$ is the gain matrix controlling the weight update and $P(t)$ is the error covariance matrix that define the search changes along the Gauss–Newton direction. Assuming that no prior information is available, $P(0)$ is usually taken $P(0) = \alpha I$, where α is an arbitrary large number.

A key issue for the performance of the algorithm is the matrix $\psi(t) = \psi(t \,|\, \hat{\vartheta}(t-1))$ defined as follows:

$$\psi(t) = -\frac{d}{d\vartheta}\varepsilon^T(t) = \frac{d}{d\vartheta}\hat{y}^T(t \,|\, \hat{\vartheta}(t-1)) \tag{25}$$

where $\psi(t)$ is a $(n \times N_M)$ matrix containing the partial derivatives of the predictor's model, that is, the network's outputs, with respect to the trainable weights

$$\psi^T(t) = \begin{pmatrix} \left(\dfrac{\partial x_1^M}{\partial \vartheta_1}\right)^T & \cdots & \left(\dfrac{\partial x_1^M}{\partial \vartheta_W}\right)^T \\ \vdots & \vdots & \vdots \\ \left(\dfrac{\partial x_{N_M}^M}{\partial \vartheta_1}\right)^T & \cdots & \left(\dfrac{\partial x_{N_M}^M}{\partial \vartheta_W}\right)^T \end{pmatrix}. \tag{26}$$

These gradients are computed using the sensitivity networks, as described in Section IV.

$[\cdot]_{D_M}$ in (22) implements a projection mechanism into the stability region. As discussed in Section IV, the gradient dynamics are identical to the forward dynamics of the recurrent network. Hence, learning stability also guarantees stable operation of the network run. The necessary and sufficient condition for the gradients to tend to zero is that the AR parts of the synaptic IIR filters should be stable. This dictates that the zeros of $[1 - A_{nm}^{(\ell)}(q-1)]$ in IIR-MLP and LAF-MLN should lie within the unit circle, which determines the stability region of the algorithm. Particularly, for the DRNN model, stability suggests that the weights should lie within the region $0 < |\nu| < 1$. Hence, for stable training of the recurrent networks, the GRPE algorithm has to be supplied with stability monitoring and a projection tool. In this paper we follow a simple approach, where

the correction term is successively reduced until the new estimates fall within the stability region.

$\mu(t)$ denotes the learning rate taking values in the range [0, 1]. Because of the gradient complexity, it scales the correction term in the weight updates and reduces the aggressiveness of GRPE during the initial training phase. $\mu(t)$ does not change at each time but after each pass through the data of the epoch (iteration). Initially, $\mu(t)$ takes a relatively small value, i.e., 0.01, thus avoiding bad estimates to be obtained. In the following iterations, as training proceeds, $\mu(t)$ is raised to unity following a user-defined profile and the GRPE takes fully over.

B. The DRPE Algorithm

The DRPE is a local learning scheme that is achieved by dividing the network weighs into G groups, at the neuron level. Each group consists of the MA and AR weights, $w_{nm(p)}^{(\ell)}$ and $v_{nm(p)}^{(\ell)}$ of the synapses pointing to a neuron. Let us consider the i-th neuron (group) described by a $M_i \times 1$ weight vector ϑ_i, so that $\vartheta^T(t) = [\vartheta_1^T(t), \ldots, \vartheta_G^T(t)]$. The effect of the i-th neuron on the network's outputs, $x_j^M, j = 1, \ldots, N_M$ is locally given by the gradient matrix

$$\psi_i(t) = \left[\left(\frac{\partial x_1^M}{\partial \vartheta_i} \right) \ldots \left(\frac{\partial x_{N_M}^M}{\partial \vartheta_i} \right) \right]. \tag{27}$$

Apparently, the above gradient is the i-th column of the global matrix $\psi^T(t)$ in (26). Consider also the covariance matrix $P(t)$ of the GRPE in block-diagonal form

$$P(t) = \mathrm{diag}(P_1(t), P_2(t), \ldots, P_G(t)). \tag{28}$$

Owing to the above weight grouping, GRPE is decomposed into a set of decoupled algorithms where the weight vector $\vartheta_i(t), i = 1, \ldots, G$, of each group is independently updated at each time t following a recursion similar to the one in (19)–(23)

$$\varepsilon(t) = y(t) - \hat{y}(t \,|\, \hat{\vartheta}(t-1)) \tag{29}$$
$$S_i(t) = \psi_i(t)^T P_i(t-1)\psi_i(t) + \lambda I \tag{30}$$
$$L_i(t) = P_i(t-1)\psi_i(t)S_i^{-1}(t) \tag{31}$$
$$\hat{\vartheta}_i(t) = [\hat{\vartheta}_i(t-1) + \mu(t)L_i(t)\varepsilon(t)]_{D_m} \tag{32}$$
$$P_i(t) = \left[P_i(t-1) - L_i(t)S_i(t)L_i^T(t)\right] / \lambda. \tag{33}$$

The recursion of each group uses the associated gradient matrices $\psi_i(t)$, and $S_i(t), L_i(t)$. Additionally, each neuron requires the storage of an individual covariance matrix $P_i(t)$ of size $M_i \times M_i$.

C. Algorithmic Complexity

Algorithmic complexity involves the computational cost and the memory demands of a particular learning algorithm. This load is measured in terms of the number of additions and multiplications required to train a network for one time-step. For simplicity, let us consider a two-layered recurrent network with N_0 inputs, one hidden layer with N_1 neurons, and N_2 neurons in the output layer. Furthermore, let $M^{(\ell)}$ denote the number of weights pertaining to the group of a neuron at the ℓth layer, $\ell = 1, 2$. As regards the IIR-MLP, we have $M^{(\ell)} = 1 + N_{\ell-1}(L^{(\ell)} + I^{(\ell)})$ while for the LAF-MLN we have $M^{(\ell)} = 1 + N_{\ell-1}L^{(\ell)} + I^{(\ell)}$. The total number of weights included in the network is $W = \sum_{\ell=1}^{2} N_\ell M^{(\ell)}$. Computational analysis shows that the computational load of the GRPE is $O(N_M W^2)$ while the storage requirements are $O(W^2)$. Additionally, the computational and storage cost of the DRPE is $O(N_2 \sum_{\ell=1}^{2} N_\ell (M^{(\ell)})^2)$ and $O(\sum_{\ell=1}^{2} N_\ell (M^{(\ell)})^2)$, respectively. Since $\sum_{\ell=1}^{2} N_\ell (M^{(\ell)})^2 \ll W^2$, it can be seen that the computational burden of DRPE is considerably smaller compared to the GRPE.

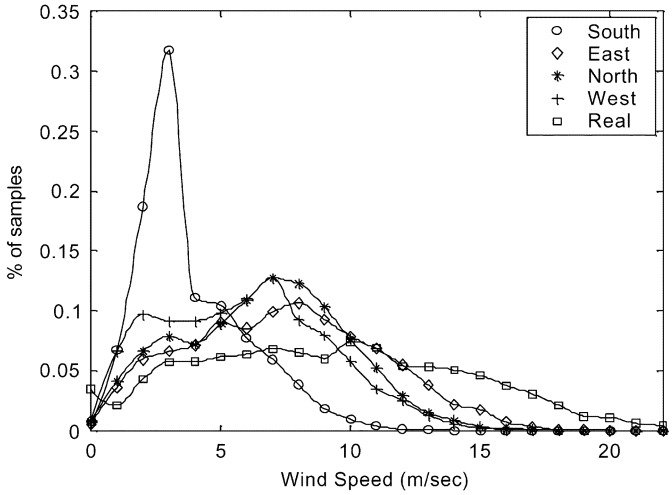

Fig. 4. Wind speed distributions of the four node predictions and the actual wind at the Rokas' park.

VI. Input Selection

As mentioned before, the recurrent neural networks were trained and evaluated using 72-h-ahead meteorological predictions at four nodes located in the vicinity of the wind park of Rokas, Crete. Although wind speed (in ms^{-1}), direction (in degrees), pressure, and temperature predictions are given by the atmospheric modeling system of SKIRON, only the real values of the wind speed and power are measured at the park site. Finally, the meteorological "prediction set" includes a linear interpolation of the node predictions derived for the exact position of the W/F.

Selection of the model's inputs is the first stage in model building. This problem has to be properly addressed having a great effect on the performance of the resulting forecast models. Considering the pool of all available candidate inputs, those variables should be selected exhibiting a significant degree of correlation with regard to the model's outputs to be forecasted. The decision as to which variables will be included as model's inputs is made on the basis of two criteria: comparison of the wind speed distributions and the degree of cross-correlation between the nodes' and actual wind values. For the Rokas' park, these criteria are shown in Fig. 4 and Table I, respectively. Fig. 4 shows that the (N, E, W) nodes have similar wind speed profiles as compared to the actual one at the park, covering the entire speed range. By contrast, the Southern node follows a different distribution with its average speed being relatively low. Moreover, Table I indicates that the Southern node has the smallest

TABLE I
AVERAGE "NODE PREDICTIONS" AND CROSS-CORRELATIONS OF THEM WITH THE REAL VALUES OF WIND SPEED AND POWER AT THE ROKAS' PARK

ROKAS	Av. Wind Speed (m/sec)	Cross-corr. with real speed	Cross-corr. with real power
S Node	3.877	0.6367	0.5573
E Node	7.429	0.7107	0.6677
N Node	6.641	0.7135	0.6859
W Node	5.947	0.7156	0.6842

degree of cross-correlation with regard to the wind speed and especially the wind power, with the rest of the nodes exhibiting an adequate correlation of almost equivalent level. This is explained by the fact that, while the (N, E, W) nodes are placed on the surface of the sea, the Southern node is located on the land, possibly on a leeward position (see Fig. 1). Therefore, the Southern node is discarded from being used as forecast model input.

The input data based on interpolated values were discarded because the linear interpolation is a very rough tool indeed to describe such a complex nonlinear system as the one under consideration. Data manipulation justifies the above decision. Finally, after extensive experimentation, it was found that inclusion of the temperature and the atmospheric pressure as inputs did not offer improved performance of the resulting models. On the contrary, the presence of two more inputs was actually slowing down the learning process. Therefore, these variables are also not included in the input set.

In view of the above discussion, six major inputs are employed as inputs to the forecast models, comprising the wind speeds and directions of the N, E, and W nodes. Before training, the data were normalized within the range $[-0.9, 0.9]$. As for the wind directions, in order for the networks to discriminate between values located around the critical point 0 or 360 degrees and establish the correct associations, they were first biased, that is, the degree axis was shifted. Particularly, the bias was set to -45 degrees, for the Rokas' park, a value decided by observing the distribution of the directions in the data.

Concluding, as can be seen from Fig. 2, the real values of wind speed and power for a day time period, involve node predictions at the input side obtained at the current, one-day and two days before. Therefore, to discriminate between the three different node-predictions corresponding to the same output data, we have introduced an additional input, the input index denoted as I_d and taking values $+0.5$, -0.5, and 0, respectively. The model's input vector contains seven input terms in total, $N_0 = 7$, and is formulated as follows:

$$x^{(0)}(t) = \left[\hat{w}_{\rm sp}^{N}(t), \hat{w}_{\rm dir}^{N}(t), \hat{w}_{\rm sp}^{E}(t), \hat{w}_{\rm dir}^{E}(t), \hat{w}_{\rm sp}^{W}(t), \hat{w}_{\rm dir}^{W}(t), I_d\right].$$

Notice that only the current values of the wind speeds and directions are used as inputs to the recurrent models at each t, leading to parsimonious forecast models with a small number of parameters. It should be mentioned that at the beginning of a day d only the meteorological node predictions are given, while the actual wind and power values are unknown (Fig. 2). Nevertheless, because of the MA and the AR parts of the synaptic filters, the output estimates are recurrently derived using past values of the inputs, as well as the network's outputs (estimates) at previous time-steps

$$\hat{y}(t) = x^{(M)}(t) = F\{\boldsymbol{x}^{(0)}(t), x^{(0)}(t-1), \ldots, \hat{y}(t-1), \hat{y}(t-2)\ldots\}.$$

The above formulation indicates that the recurrent models fulfill all the requirements imposed by an efficient forecast model, as suggested by (2). Our method combines the time-series approach [4], [5], and the atmospheric modeling, simultaneously. On the other hand, in the absence of future target data, when conventional static (memoryless) NNs are employed for multi-step ahead prediction, we usually follow two approaches. According to the first approach, a separate model is developed, providing the wind forecasts at each time-step ahead. The second approach consists of feeding back the model's output estimates as explicit inputs through tapped-delay lines. In either case, we are faced with two drawbacks. First, the order of the past values to be introduced to the network is unknown in practice. Secondly, the inclusion of additional inputs aggravates the parametric complexity of the models.

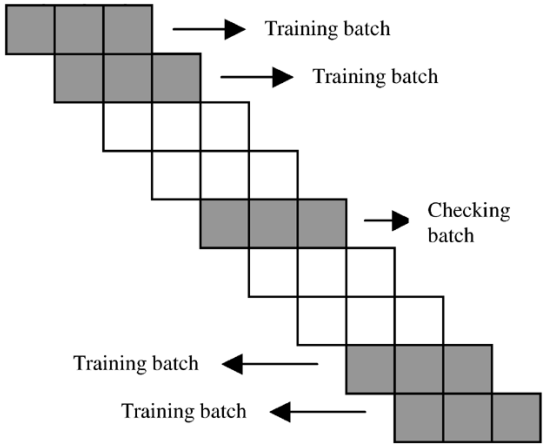

Fig. 5. Selection of the training and checking data patterns. Each square denotes a 24-hour batch.

After having determined the model inputs, the training and testing data patterns are created, as shown in Fig. 5. The patterns are arranged in data batches containing 72 hourly node predictions (inputs), along with the 72 respective values of wind speed or power measurements. The data batches are selected in such a way so that neither the training nor the checking data sets are overlapping, thus guaranteeing complete independence between these two sets. Finally, in order to ensure continuity of the network's states with the ones of the next 72-hour data batch, we introduced in front of each batch the 24-hour data pairs of the previous day. These extra 24 h are not used for training; they are simply passed through the network to develop the proper states.

VII. SIMULATION RESULTS

The available data are divided into training and a checking data set, composed of 3264 and 960 patterns, respectively. The training data set is used for training of the recurrent models using the learning algorithms suggested in Section V.

Moreover, the checking data set is used to evaluate the forecast performance of the resulting models. For each recurrent

network type, two separate forecast models are developed, providing at the beginning of each day, 72-h-ahead forecasts of the wind speed and power at the park. On-line training is carried out on the data batches for 400 epochs.

IIR-MLP networks with two hidden layers and seven neurons per layer are selected. Both MA and AR parts of order 3 are considered for the IIR synaptic filters in the hidden neurons. Especially for the output neuron, an AR part of order 5 is chosen. This allows the model to learn more efficiently the dependence of the current output with regard to its past. Structures with similar number of parameters are chosen for the other two recurrent networks used. Particularly, a network with two hidden layers and eight neurons per layer is considered for the LAF-MLN. The order of the MA parts is set to 4 while the AR parts are the same as in the case of IIR-MLP models. Finally, a DRNN model with one hidden layer composed of 32 self-recurrent neurons is selected.

In order to validate our input selection, we considered three scenarios (models), in which only the one (W), two (N, W), or three (N, W, E) most correlated nodes are used as inputs to the networks. Hence, the respective networks have three, five, and seven inputs, respectively: the selected nodes' predictions of the speed and direction, and the input index.

The network weights $w_{nm(p)}^{(\ell)}$ are randomly selected initially in the range $[-0.5, 0.5]$ while $v_{nm(p)}^{(\ell)}$, involved in the AR parts of the synaptic filters, are initialized so that the roots of the resulting polynomial $[1 - A_{nm}^{(\ell)}(q-1)]$ lie inside the unit circle, as required for stable operation of the learning algorithm. As an activation function, the hyperbolic tangent, $\mathrm{sgm}(x) = \tanh(x)$, is used in our experiments. In order to avoid excessive errors during the training stage caused by bad initial estimates of the weights, the learning rate $\mu(t)$ was set initially to a small value, $\mu(0) = 0.1$. In the following, as learning proceeds, $\mu(t)$ is gradually increased to unity, following the formula $\mu(t) = \xi \times \mu(t-1) + (1-\xi)$, where ξ is set to 0.8. The correlation matrix $P(t)$ is initialized as $P(0) = 500 \times I$, where I is the identity matrix with size equal to the one of the weight vector.

Finally, during learning, stability monitoring of the RPE algorithms is continuously performed. Assuming that the current estimates lie outside the stability region, the projection mechanism is activated. Accordingly, the correction term is successively multiplied by a factor of 0.2 until the revised weight estimates obtained fulfill the stability conditions for each neuron and network type.

In order to justify the benefit gained by using local recurrent NNs, apart from the IIR-MLP, the LAF-MLN and the DRNN, two additional static models are examined for comparison, namely, a static MLP and a FIR neural network where the connection weights are realized by linear FIR filters [14]. The FIR-NNs are functionally equivalent to static MLPs, although with better modeling capabilities since due to the FIR synapses, past values of the inputs are also taken into consideration. The networks had again three, five, or seven inputs, depending on the node predictions being used.

To establish a fair comparison basis, the structure of the competing networks is selected so that they contain approximately the same number of parameters as the recurrent networks. Particularly, a static MLP is chosen with two hidden layers and 20 neurons per layer, and a FIR-NN with two hidden layers and linear FIR filters of sixth order. Furthermore, after the necessary modifications to the computation of the network gradients, all forecast models are trained by means of the suggested GRPE and DRPE algorithms. Under these conditions, the models are evaluated in terms of representation power and their capabilities to produce efficient multistage forecasts. Nevertheless, an additional case is considered where the static MLP is trained using the conventional BP algorithm.

Based on the data available for the Rokas' park, for each input case (3, 2, 1 input nodes) and network type, an exhaustive set of experiments is carried out. As a measure to evaluate the performance of the forecast models the mean absolute (MAE) and the root mean square error (rmse) are used. The best results achieved for each case are cited in Tables II and III, where the models are trained by GRPE and the DRPE, respectively.

TABLE II
MAE AND RMSE FOR WIND POWER AND SPEED FORECASTS OBTAINED BY THREE MODEL TYPES INCLUDING THREE (N, W, E), TWO (N, W), AND ONE (W) INPUT NODES, AND TRAINED BY THE GRPE ALGORITHM

	Inputs	Wind Power		Wind Speed	
		MAE	RMS	MAE	RMS
IIR-MLP	7	1.2117	1.5263	1.9755	2.7211
	5	1.3085	1.6925	2.2088	2.8241
	3	1.4806	1.9061	2.3395	2.9844
LAF-MLN	7	1.3218	1.6610	1.9994	2.6794
	5	1.4095	1.7817	2.0468	2.7210
	3	1.4332	1.8139	2.2151	2.9304
DRNN	7	1.3428	1.7144	2.0456	2.7842
	5	1.4235	1.8652	2.1982	2.9812
	3	1.4770	1.8781	2.3225	3.0965
FIR	7	1.3741	1.7601	2.1270	2.8312
	5	1.4315	1.8078	2.2144	3.0121
	3	1.4613	1.8505	2.2146	3.0209
Static MLP (GRPE)	7	1.3880	1.7905	2.1815	2.8021
	5	1.5442	1.9223	2.2957	2.9679
	3	1.5801	2.0202	2.3703	3.0845
Static MLP (BP)	7	1.6143	2.1980	2.5037	3.1262
	5	1.7917	2.2780	3.0003	3.6330
	3	2.1406	2.5463	3.2461	3.9304

TABLE III
MAE AND RMSE FOR WIND POWER AND SPEED FORECASTS OBTAINED BY THREE MODEL TYPES INCLUDING THREE (N, W, E), TWO (N, W), AND ONE (W) INPUT NODES, AND TRAINED BY THE DRPE ALGORITHM

	Inputs	Wind Power		Wind Speed	
		MAE	RMS	MAE	RMS
IIR-MLP	7	1.3305	1.7218	2.1678	2.8136
	5	1.4466	1.8932	2.2137	2.8539
	3	1.4741	1.9105	2.3194	2.9398
LAF-MLN	7	1.3963	1.7579	2.0205	2.7365
	5	1.4214	1.7950	2.1278	2.8271
	3	1.4750	1.8656	2.2140	2.8756
DRNN	7	1.4332	1.8471	2.1961	2.9251
	5	1.4864	1.9406	2.2720	3.0627
	3	1.5403	1.9545	2.3873	3.1101
FIR	7	1.4815	1.9216	2.2417	2.9948
	5	1.5042	1.9575	2.3022	3.0548
	3	1.5616	2.0217	2.4201	3.1312
Static MLP	7	1.4693	1.9029	2.2124	2.9762
	5	1.5409	1.9930	2.3305	3.0643
	3	1.5931	2.0465	2.4192	3.1585

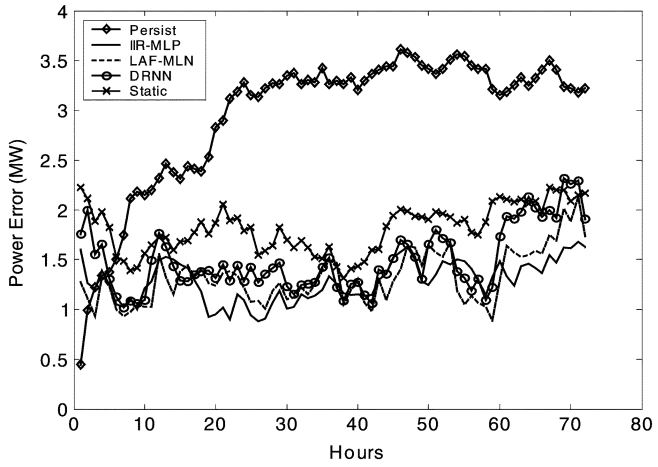

Fig. 6. Power forecast errors (MAE) for a horizon of 72 h obtained by the IIR-MLP, LAF-MLN, DRNN, and the static MLP (trained by BP) with three input nodes, along with the errors of the persistent method.

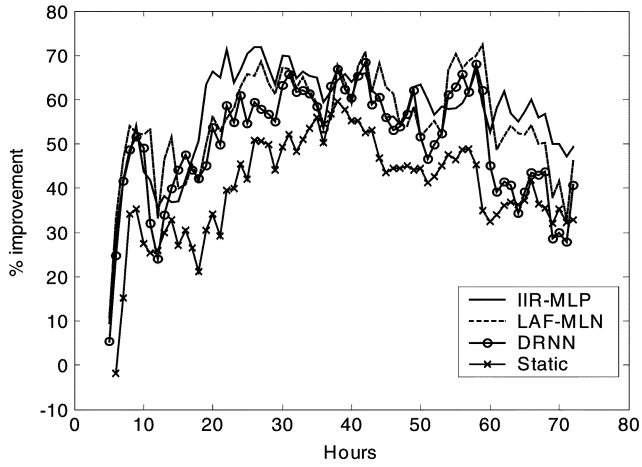

Fig. 7. Percentage improvement over the persistent method of the power forecasts achieved by IIR-MLP, LAF-MLN, DRNN, and the static MLP (trained by BP) with three input nodes.

It can be seen that the wind speed and power performance is improved when additional nodes are included as inputs. The best results are attained for each network type when the complete models are considered, including all three nodes (N, E, W). Since the meteorological predictions at the three most correlated nodes contain almost an equivalent amount of information, the above observation verifies experimentally our input selection.

For the models trained by GRPE (Table II) it can be seen that the recurrent forecast models exhibit consistently better performance compared to the static models, FIR-NN and the static MLP, thus justifying their use for multiple-step wind forecasting. Owing to the richness of the network architecture, IIR-MLP shows the best performance, outperforming the LAF-MLN and the DRNN models in all cases. Particularly, IIR-MLP provides for the wind power and speed a MAE (rmse) of 1.2117 MW (1.5263) and 1.9755 ms^{-1} (2.7211). Regarding the power forecasts, it outperforms the FIR-NN and the static MLP by 11.82% and 12.7% in terms of MAE. Furthermore, for the speed forecasts, an improvement of 7.12% and 9.44% is obtained, respectively. Notice that the static models in Table II are trained using an enhanced learning scheme, the GRPE algorithm. Assuming that the MLP is trained by the conventional BP method, a considerably larger improvement is achieved. In that case, the IIR-MLP outperforms the static MLP by 24.94% and 21.10% for the wind power and speed, respectively.

Fig. 6 depicts the forecasting errors (MAE) of the wind power, for the complete models of the three recurrent networks trained by GRPE and the static MLP, along with the errors obtained by the persistent method. Fig. 7 shows the percentage improvement in wind power forecasting of each network model over the persistent method, on the basis of MAE criterion. In view of the above figures, the following observations are in order.

— The persistent error is almost monotonically increasing when larger time-steps are considered in the future. This behavior is continued for the first 50 steps ahead, settling to a high error level for the rest of the time horizon. Nevertheless, good estimates are obtained for the first few time-steps as expected. This is the time range (up to 6–8 h ahead) where it is suggested to use short-term models (or a coupling of them with long-term models) in order to outperform persistent forecast.

— Compared to the static MLP, the recurrent models exhibit, consistently, the best forecasting errors for all time-steps ahead; the best performance is shown by IIR-MLP. In contrast to the persistent method, the errors move in the vicinity of the MAE value (1.2117 MW) for the entire time horizon. This indicates the capability of the recurrent models to produce robust multi-step ahead predictions. As a result, considerable improvement is attained over the persistent forecasts. Particularly, an average improvement of over 50% is obtained for time-steps larger that 20, although smaller improvement is shown for shorter time lags. Similar observations are also valid for the wind speed, as shown in Figs. 8 and 9 where the forecasting errors and improvement over the persistent method are given.

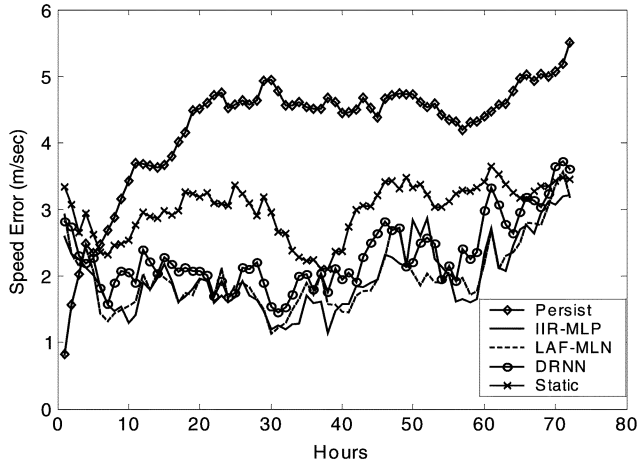

Fig. 8. Speed forecast errors (MAE) for a horizon of 72 h obtained by the IIR-MLP, the LAF-MLN, the DRNN, and the static MLP (trained by BP) with three input nodes, along with the errors of the persistent method.

The ability of the model to learn the process dynamics and provide efficient forecasts is demonstrated in Fig. 10. A typical 72-h curve belonging to the checking data set is considered,

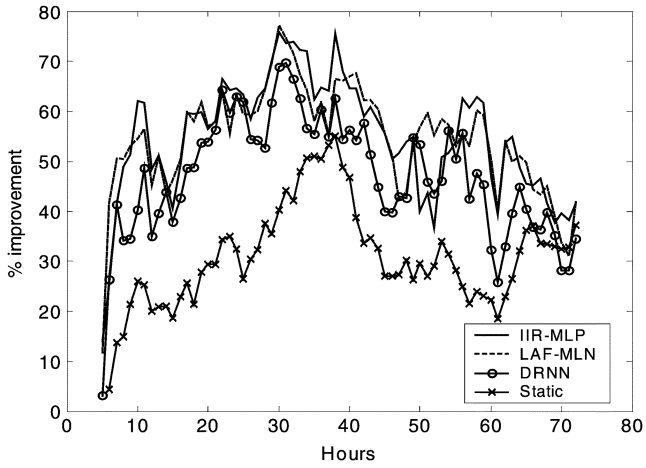

Fig. 9. Percentage improvement over the persistent method of the speed forecasts achieved by IIR-MLP, the LAF-MLN, the DRNN, and the static MLP (trained by BP) with three input nodes.

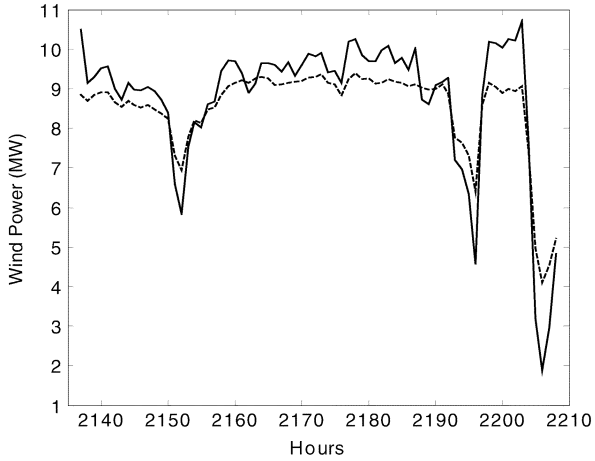

Fig. 10. Real wind power (solid line) and predicted wind power (dashed line) in MW, for a typical power forecast curve of the checking data set.

including the actual values of the wind power at the park and the outputs of an IIR-MLP model. As shown, the forecasts are good, following the trends of the real power closely. Similarly, in Fig. 11 a typical curve is plotted including wind speed forecasts on the park site, with the nodes' respective forecasts provided by the SKIRON system.

From the results shown in Table III it can be concluded that the forecast models trained by the simplified local algorithm, the DRPE, exhibit slightly inferior performance as contrasted to the ones obtained by the GRPE (Table II). Notice however, that as revealed from the complexity analysis, DRPE has considerably less requirements than GRPE, in terms of computational cost and storage needs. Nevertheless, the overall picture remains the same, that is, the recurrent models outperform the static forecast models, with the best results achieved by the IIR-MLP.

VIII. Conclusion

Three local recurrent neural networks are employed in this paper, providing 72 time-steps ahead forecasts of the wind speed and power at the Rokas' wind park on the Greek island of Crete. Forecasting is based on meteorological data given at four

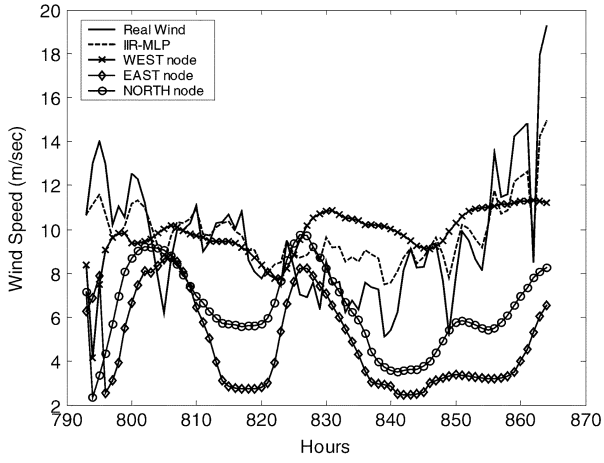

Fig. 11. Real wind speed (solid line) and predicted wind speed (dashed line) in meters per second (m/s), for a typical speed forecast curve of the checking data set, along with the respective predictions for three surrounding nodes.

nodes nearby the park site. Two novel learning algorithms are introduced for the training of the recurrent forecast models, the GRPE and the DRPE, that have considerably smaller computational and storage requirements. Extensive experimentation and model comparison reveals the effectiveness of the suggested learning methods. Moreover, it is shown that the recurrent forecast models outperform the static rivals in terms of forecast errors and the improvement gained over the persistent method.

Appendix

In view of the multilayer structure of the IIR-MLP, we can distinguish three distinct cases as described below.

Case 1: Gradients of the neuron's output $x_n^{(\ell)}[t]$ with respect to synaptic weights $\nu_{nm}^{(\ell)}$.

First, from (2) and (3), we have

$$\frac{\partial^+ x_n^{(\ell)}[t]}{\partial \nu_{nm}^{(\ell)}} = \mathrm{sgm}'\{s_n^{(\ell)}[t]\} \cdot \frac{\partial^+ y_{nm}^{(\ell)}[t]}{\partial \nu_{nm}^{(\ell)}}. \tag{A.1}$$

Applying the ordered derivatives forward chain rule with respect to w's, we finally get

$$\frac{\partial^+ y_{nm}^{(\ell)}[t]}{\partial w_{nm(p)}^{(\ell)}} = x_m^{(\ell-1)}[t-p] + \sum_{r=1}^{I_{nm}^{(\ell)}} \nu_{nm(p)}^{(\ell)} \frac{\partial^+ y_{nm}^{(\ell)}[t-r]}{\partial w_{nm(p)}^{(\ell)}} \tag{A.2}$$

and following an adaptive filter notation as

$$\frac{\partial^+ y_{nm}^{(\ell)}[t]}{\partial w_{nm(p)}^{(\ell)}} = \frac{x_m^{(\ell-1)}[t-p]}{[1 - A_{nm}^{(\ell)}(q^{-1})]} \tag{A.3}$$

where $\partial y/\partial w$ and $\partial^+ y/\partial^+ w$ denote an ordinary and an ordered derivative. Similarly, differentiation with respect to v's leads to the following relations:

$$\frac{\partial^+ y_{nm}^{(\ell)}[t]}{\partial u_{nm(p)}^{(\ell)}} = y_{nm}^{(\ell)}[t-p] + \sum_{r=1}^{I_{nm}^{(\ell)}} u_{nm(p)}^{(\ell)} \frac{\partial^+ y_{nm}^{(\ell)}[t-r]}{\partial u_{nm(p)}^{(\ell)}} \tag{A.4}$$

$$\frac{\partial^+ y_{nm}^{(\ell)}[t]}{\partial u_{nm(p)}^{(\ell)}} = \frac{y_{nm}^{(\ell-1)}[t-p]}{[1 - A_{nm}^{(\ell)}(q^{-1})]}. \tag{A.5}$$

Case 2: Gradients of the neuron's output $x_q^{(\ell+1)}[t]$ of the $(\ell+1)$th layer with respect to synaptic weights $\nu_{nm}^{(\ell)}$ of the ℓth layer.

Using the network's architecture and the forward chain formula, we get

$$\frac{\partial^+ y_{qn}^{(\ell+1)}[t]}{\partial \nu_{nm(p)}^{(\ell)}} = \sum_{r=0}^{(L_{qn}^{(\ell+1)}-1)} w_{qn(r)}^{(\ell+1)} \frac{\partial^+ x_n^{(\ell)}[t-r]}{\partial \nu_{nm(p)}^{(\ell)}} + \sum_{r=1}^{I_{qn}^{(\ell+1)}} \nu_{qn\,(r)}^{(\ell+1)} \frac{\partial^+ y_{qn}^{(\ell+1)}[t-r]}{\partial \nu_{nm(p)}^{(\ell)}} \quad \text{(A.6)}$$

or

$$\frac{\partial^+ y_{qn}^{(\ell+1)}[t]}{\partial \nu_{nm(p)}^{(\ell)}} = \frac{B_{qn}^{(\ell+1)}(q^{-1})}{\left\{1 - A_{qn}^{(\ell+1)}(q^{-1})\right\}} \frac{\partial^+ x_n^{(\ell)}[t]}{\partial \nu_{nm(p)}^{(\ell)}}. \quad \text{(A.7)}$$

The required gradients are derived through

$$\frac{\partial^+ x_q^{(\ell+1)}[t]}{\partial \nu_{nm(p)}^{(\ell)}} = \mathrm{sgm}'\left\{s_q^{(\ell+1)}[t]\right\} \cdot \frac{\partial^+ y_{qn}^{(\ell+1)}[t]}{\partial \nu_{nm(p)}^{(\ell)}}. \quad \text{(A.8)}$$

Case 3: Gradients of the neuron's output $x_u^{(\ell+i)}[t]$ of the $(\ell+i)$th, $i = 2, 3, \ldots$ layer with respect to synaptic weights $\nu_{nm}^{(\ell)}$ of the ℓth layer.

Based on the network's structure, we have

$$\frac{\partial^+ y_{uq}^{(\ell+\iota)}[t]}{\partial \nu_{nm(p)}^{(\ell)}} = \sum_{r=0}^{(L_{uq}^{(\ell+\iota-1)}-1)} w_{uq(r)}^{(\ell+\iota)} \frac{\partial^+ x_q^{(\ell+\iota-1)}[t-r]}{\partial \nu_{nm(p)}^{(\ell)}} + \sum_{r=1}^{I_{uq}^{(\ell+\iota-1)}} \nu_{uq(r)}^{(\ell+\iota)} \frac{\partial^+ y_{uq}^{(\ell+\iota)}[t-r]}{\partial \nu_{nm(p)}^{(\ell)}}. \quad \text{(A.9)}$$

The above equation can be rewritten as

$$\frac{\partial^+ y_{uq}^{(\ell+i)}[t]}{\partial \nu_{nm(p)}^{(\ell)}} = \frac{B_{uq}^{(\ell+i)}(q^{-1})}{\left\{1 - A_{uq}^{(\ell+i)}(q^{-1})\right\}} \frac{\partial^+ x_q^{(\ell+i-1)}[t]}{\partial \nu_{nm(p)}^{(\ell)}}. \quad \text{(A.10)}$$

The required derivatives for the neuron outputs are

$$\frac{\partial^+ s_u^{(\ell+i)}[t]}{\partial \nu_{nm(p)}^{(\ell)}} = \sum_{q=1}^{N_{\ell+i-1}} \frac{B_{uq}^{(\ell+i)}(q^{-1})}{\left\{1 - A_{uq}^{(\ell+i)}(q^{-1})\right\}} \frac{\partial^+ x_q^{(\ell+i-1)}[t]}{\partial \nu_{nm(p)}^{(\ell)}} \quad \text{(A.11)}$$

where

$$\frac{\partial^+ x_u^{(\ell+i)}[t]}{\partial \nu_{nm(p)}^{(\ell)}} = \mathrm{sgm}'\left\{s_u^{(\ell+i)}[t]\right\} \cdot \frac{\partial^+ s_u^{(\ell+i)}[t]}{\partial \nu_{nm(p)}^{(\ell)}}. \quad \text{(A.12)}$$

ACKNOWLEDGMENT

SKIRON is a weather forecasting system initially developed for the Hellenic Meteorological Service based on the Eta/NCEP model. The meteorological numerical predictions are now produced on a daily basis by the Atmospheric Modeling and Weather Forecasting Group (AM&WFG) of the University of Athens, Athens, Greece [24]. The authors especially wish to thank Prof. G. Kallos and Dr. P. Katsafados for their collaboration. Actual data from the Rokas W/F at Crete are provided by Public Power Corporation of Greece. All data were gathered and given to the authors under the frame of the MORE-CARE project supported by the European Commission.

REFERENCES

[1] E. A. Bossanyi, "Short-term wind prediction using Kalman filters," *Wind Eng.*, vol. 9, no. 1, pp. 1–8, 1985.

[2] J. O. G. Tande and L. Landberg, "A 10 sec. forecast of wind turbine output with neural networks," in *Proc. 4th European Wind Energy Conf. (EWEC'93)*, Lübeck-Travemünde, Germany, 1993, pp. 747–777.

[3] G. C. Contaxis and J. Kabouris, "Short-term scheduling in wind/diesel autonomous system," *IEEE Trans. Power Syst.*, vol. 6, no. 3, pp. 1161–1167, Aug. 1991.

[4] G. N. Kariniotakis, G. S. Stavrakakis, and E. F. Nogaret, "Wind power forecasting using advanced neural networks models," *IEEE Trans. Energy Convers.*, vol. 11, no. 4, pp. 762–767, Dec. 1996.

[5] A. More and M. C. Deo, "Forecasting wind with neural networks," *Marine Structures*, vol. 16, pp. 35–49, 2003.

[6] L. Landberg and S. J. Watson, "Short-term prediction of local wind conditions," *Boundary-Layer Meteorol.*, vol. 70, p. 171, 1994.

[7] L. Landberg, A. Joensen, G. Giebel, H. Madsen, and T. S. Nielsen, "Short-term prediction toward the 21st century," in *Proc. British Wind Energy Association*, vol. 21, Cambridge, U.K., 1999.

[8] L. Landberg and A. Joensen, "A model to predict the output from wind farms—An update," in *Proc. British Wind Energy Association*, vol. 20, Cardiff, Wales, U.K., 1998.

[9] S. J. Watson, G. Giebel, and A. Joensen, "The economic value of accurate wind power forecasting to utilities," in *Proc. EWEC99*, Nice, France, 1999, pp. 1109–1012.

[10] T. S. Nielsen and H. Madsen, "Experiences with statistical methods for wind power prediction," in *Proc. EWEC99*, Nice, France, 1999, pp. 1066–1069.

[11] E. Akylas, "Investigation of the effects of wind speed forecasts and economic Evaluation of the increased penetration of wind energy for the island of Crete," in *Proc. EWEC99*, Nice, France, 1999, pp. 1074–1077.

[12] G. Kariniotakis, D. Mayer, J. A. Halliday, A. G. Dutton, A. D. Irving, R. A. Brownsword, P. S. Dokopoulos, and M. C. Alexiadis, "Load, wind, and hydro power forecasting functions of the more-care EMS system," in *Proc. Med Power 2002*, Athens, Greece, Nov. 2002.

[13] A. C. Tsoi and A. D. Back, "Locally recurrent globally feedforward networks: A critical review of architectures," *IEEE Trans. Neural Netw.*, vol. 5, no. 3, pp. 229–239, May 1994.

[14] E. A. Wan, "Temporal backpropagation for FIR neural networks," in *Proc. Int. Joint Conf. Neural Networks,*, vol. 1, 1990, pp. 575–580.

[15] S. Haykin, *Neural Networks: A Comprehensive Foundation*. New York: IEEE Press, 1994.

[16] P. Frasconi, M. Gori, and G. Soda, "Local feedback multilayered networks," *Neural Comput.*, vol. 4, pp. 120–130, 1992.

[17] C.-C. Ku and K. Y. Lee, "Diagonal recurrent neural networks for dynamic systems control," *IEEE Trans. Neural Netw.*, vol. 6, no. 1, pp. 144–156, Jan. 1995.

[18] P. J. Werbos, "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences," Ph.D. dissertation, Committee on Appl. Math., Harvard Univ., Cambridge, MA, 1974.

[19] E. A. Wan and F. Beaufays, "Diagrammatic derivation of gradient algorithms for neural networks," *Neural Comput.*, vol. 8, pp. 182–201, 1996.

[20] R. J. Williams and J. Peng, "An efficient gradient-based algorithm for on-line training of recurrent network structures," *Neural Comput.*, vol. 2, pp. 490–501, 1990.

[21] L. Ljung and T. Söderström, *Theory and Practice of Recursive Identification*, Cambridge, U.K.: MIT Press, 1983.

[22] S. Shah, F. Palmieri, and M. Datum, "Optimal filtering algorithms for fast learning in feedforward neural networks," *Neural Netw.*, vol. 5, pp. 779–787, 1992.

[23] G. V. Puskorius and L. A. Feldkamp, "Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 279–297, Mar. 1994.

[24] G. Kallos, S. Nickovic, A. Papadopoulos, and P. Katsafados, "The SKIRON forecasting system and its capability to predict extreme weather events in the Mediterranean," in *Proc. 7th Int. Symp. Natural and Man-Made Hazards (HAZARDS-98)*, Chania, Greece, May 1998.

Thanasis G. Barbounis was born in Lamia, Greece, in 1977. He graduated in electrical engineering in 1999 from the Aristotle University of Thessaloniki, Thessaloniki, Greece, where he is currently pursuing the Ph.D. degree.

His research interests include artificial neural networks, fuzzy logic systems, and modeling of nonlinear systems.

John B. Theocharis (M'90) graduated in electrical engineering in 1980 and received the Ph.D. degree in 1985 , both from the Aristotle University of Thessaloniki, Thessaloniki, Greece.

He is currently an Associate Professor in the Department of Electronic and Computer Engineering, Aristotle University of Thessaloniki. His research activities include fuzzy systems, neural networks, adaptive control, and modeling of complex nonlinear systems.

Minas C. Alexiadis was born in Thessaloniki, Greece, in July 1969. He received the Dipl. Eng. degree in 1994 and the Ph.D. degree in 2003, both from the Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Thessaloniki, Greece.

His research interests include renewable energy sources and artificial intelligence applications in power systems.

Petros S. Dokopoulos (M'77) was born in Athens, Greece, in September 1939. He received the Dipl. Eng. degree from the Technical University of Athens, Athens, Greece, in 1962 and the Ph.D. degree from the University of Brunswick, Brunswick, Germany, in 1967.

He was with the Laboratory for High Voltage and Transmission, University of Brunswick (1962–1967), the Nuclear Research Center at Julich, Julich, Germany (1967–1974), and the Joint European Torus (1974–1978). Since 1978, he has been a Full Pofessor at the Department of Electrical Engineering, Aristotle University of Thessaloniki. He has worked as a Consultant to Brown Boveri and Cie, Mannheim, Germany, to Siemens, Erlagen, Germany, to Public Power Corporation, Greece, and to National Telecommunication Organization, Greece. His scientific fields of interest are dielectrics, power switches, generators, power cables, alternative energy sources, transmission, and distribution and fusion.