Introducing Distributed Learning Approaches in Wind Power Forecasting

Pierre Pinson, *Senior Member, IEEE*
Technical University of Denmark, Department of Electrical Engineering
Center for Electric Power and Energy
Kgs. Lyngby, Denmark
Email: ppin@elektro.dtu.dk

***Abstract*—Renewable energy forecasting is now of core interest to both academics, who continuously propose new forecast methodologies, and forecast users for optimal operations and participation in electricity markets. In view of the increasing amount of data being collected at power generation sites, thanks to substantial deployment of generating capacities and increased temporal resolution, it may now be possible to build large models accounting for all space-time dependencies. This will eventually allow to significantly improve the quality of short-term renewable power forecasts. However, in practice, it is often the case that operators of these generation sites do not want to share their data due to competitive interests. Consequently, approaches to privacy-preserving distributed learning are proposed and investigated here. These permit to take advantage of all potential data collected by others, without having to ever share any data, by decomposing the original large learning problem into a number of small learning problems that can be solved in a decentralized manner. As an example, emphasis is placed on Lasso-type estimation of autoregressive models with offsite observations. Different applications on medium to large datasets in Australia (22 wind farms) and France (85 wind farms) are used to illustrate the interest and performance of our proposal.**

***Index Terms*—Distributed optimization, time-series analysis, vector autoregressive model, sparse estimation, short-term forecasting**

I. Introduction

TO SUPPORT THE large-scale deployment of renewable energy generation capacities, forecasting has received increasing and substantial focus over the last few decades. Today, one can find numerous approaches to wind and solar power forecasting. These may be proposed in both deterministic and probabilistic framework, while focused on very-short to longer lead times. While most of these approaches are based on limited input data, there is today a tendency in accounting for more and more data as input (offsite information, exogenous variables from remote sensing and weather forecasts, etc.). The interested reader is referred to recent reviews on renewable energy forecasting in, e.g., [1], [2]. Wind power forecasting might be seen as more mature than for the case of solar energy, owing to an earlier start and to the specifics of some of the necessary developments, e.g., related to clear-sky models and cloud passages for solar power. This hence justify our preference to describe our proposal within the frame of wind power forecasting. However, the most basic idea of distributed learning may be readily translated to the case of solar power, with similar aims and techniques.

Among the many modern challenges in wind power forecasting [3], a methodological one is that of considering an ever-increasing number of sites. While most works have looked at how to predict wind power generation at a given site or for a given portfolio, only few have considered to directly and jointly predict at multiple sites, by also accounting for the spatio-temporal correlation in their power generation. Notable recent examples include [4]–[6] (for both wind and solar power, respectively). Eventually, one may want to obtain forecasts for tens of sites for the example of portfolio management, and possibly for 100s or 1000s of sites for network management. A practical limitation relates to the fact that owners and operators of all these sites may not be keen on sharing their data, due to competitive commercial and industrial interests. To jointly consider these methodological and practical challenges, we describe here a first proposal for privacy-preserving distributed learning in wind power forecasting. In short, based on the obvious fact such that a learning problem is formulated as an optimization problem, we consider modern techniques of optimization based on decomposition to split a large centralized learning problem into a large set of distributed (and small) learning problems. A direct advantage is that, by doing so, the data collected at each and every site never has to be exchanged with other wind farms and with the centralized operator. This is while insuring that the solution of the distributed learning approach is very close to that of a centralized learning problem for which all information would be readily available. Another advantage is that of dampening computational costs for the centralized operator, since computation is also distributed at all sites considered.

To introduce our distributed learning proposal, we limit ourselves to very-short term forecasting (i.e., less than 1 hour ahead), which only uses recent measurements at all locations of interest. Extension and generalization to the case of further lead times, with extra input from remote-sensing and weather forecasts, may be considered in future works, within the same methodological framework. Our general approach to model ling and forecasting is firstly described in Section II. Section III describes the proposed method for distributed learning, which is directly inspired by the alternating direction method of multipliers extensively described in [7], where the overall learning problem is decomposed on a per-feature basis. Application results are gathered in Section IV for two

alternative setups in Australia (22 wind farms) and France (85 wind farms). The paper ends with a set of conclusions and perspectives for future work in Section V.

II. Modeling and Forecasting Framework

Let us place ourselves within a modeling and forecasting framework similar to that used in [4], [8]. Wind power generation data is being collected at a number m of power generation sites. We use $x_{j,t}$ to denote the power measurement at site s_j, $j = 1, \ldots, m$ and time t, $t = 1, \ldots, T$, where T is the number of time steps in the dataset. Write Ω the overall set of wind farms, and Γ the set of time indices. Power measurements at each and every site are normalized by the nominal capacity of the respective wind farms, P_j, $j = 1, \ldots, m$. Besides, we assume that a given actor of the power system, being the operator of a single wind farm or of a portfolio of wind farms, or alternatively the system operator, cares about the response variable y_t at time t. For simplicity, we will refer to him as the *central agent* in the following. The response variable may readily be the power production at a given site s_j of interest for the single wind farm operator, i.e., $y_t = x_{j,t}, \forall t$. For the case of a portfolio Ω_p of m_p wind farms, $m_p \leq m$, defined as

$$\Omega_p = \{s_{p,1}, s_{p,2}, \ldots, s_{p,m_p}\}, \quad \Omega_p \in \Omega \tag{1}$$

the power generation y_t at time t is given as a weighted average of the power generation of the individual farms,

$$y_t = \frac{1}{\sum_{S_j \in \Omega_p} P_j} \sum_{s_j \in \Omega_p} P_j x_{j,t}, \quad \forall t \tag{2}$$

In the case of the system operator, the portfolio consists in the full set of wind farms, i.e., $\Omega_p = \Omega$. This is a special case though, since we would assume that the system operator does not have access and control over the data of these wind farms. If considering the limiting case of a single wind farm, Ω_p includes that wind farm only.

In practice, the central agent may have direct access to its own data (being only a portfolio average in the case of the system operator), while having made agreements for distributed learning with a set Ω_a of m_a wind farms, $m_a \leq m$,

$$\Omega_a = \{s_{a,1}, s_{a,2}, \ldots, s_{a,m_a}\}, \quad \Omega_a \in \Omega \tag{3}$$

For a single wind farm or a portfolio operator, one has $\Omega_a \in \Omega \setminus \Omega_p$, which is not the case if considering the system operator, for which $\Omega_a = \Omega_p$. We refer to those wind farms with whom distributed learning agreements are made as the *contracted agents*. The overall architecture of the distributed learning problem we are looking at is depicted in Figure 1.

Based on this organizational setup, short-term forecasting is to be based on a time-series forecasting approach based on recent measurements. For simplicity, and since it is often the most relevant setup (see, e.g., [4], [8]), we restrict ourselves to autoregressive models accounting for local and offsite information. This reads

$$y_t = \beta_0 + \sum_{s_j \in \Omega_p} \sum_{\tau=1}^{l} \beta_{j,\tau} x_{j,t-\tau} + \sum_{s_j \in \Omega_a} \sum_{\tau=1}^{l} \beta_{j,\tau} x_{j,t-\tau} + \varepsilon_t \tag{4}$$

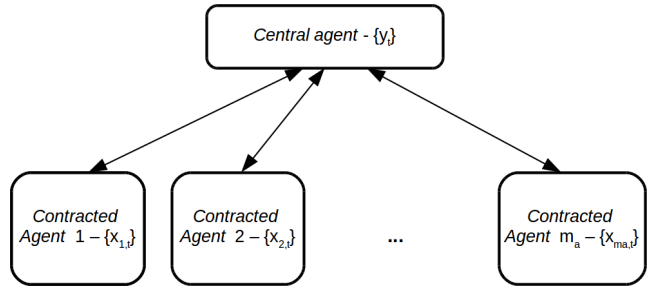

Fig. 1: Architecture of the distributed learning problem, with a central agent and his response data $\{y_t\}$, as well as a number of contracted agents and their data $\{x_{j,t}\}$.

where τ is a lag variable, between 1 and l the maximum lag of the autoregressive model. Alternative versions of this model may be considered, for instance with different lags for the various sites, while possibly overlooking the constant term β_0. Since being linear, this model may be written in a more compact form as

$$y_t = \boldsymbol{\beta} \mathbf{x}_t + \varepsilon_t \tag{5}$$

where $\boldsymbol{\beta}$ is a vector gathering all model parameters and $\mathbf{x}_t$ is for the vector of ordered values for all corresponding explanatory variables at time t. In the form of (4), the dimension of $\boldsymbol{\beta}$ is $l(m_a + m_p) + 1$.

Typically, one would obtain an estimate $\hat{\boldsymbol{\beta}}$ of the parameter vector in a centralized fashion since gathering all information and solving this learning problem as a large optimization problem. Then, using that estimate $\hat{\boldsymbol{\beta}}$ and the latest information on explanatory variables, a forecast issued at time t for time $t+1$ is readily given by

$$\hat{y}_{t+1|t} = \hat{\boldsymbol{\beta}} \mathbf{x}_{t+1} \tag{6}$$

III. Approach to Distributed Learning

Assuming that one deals with a large number of wind farms, the size of the parameter vector $\boldsymbol{\beta}$ will grow rapidly. Consequently, it may be preferable to consider a Lasso-type (for "least absolute shrinkage and selection operator") estimator instead of a more classical Least-Squares (LS) one. An advantage is that it will force most values of $\boldsymbol{\beta}$ to be 0s, hence allowing for some form of variable selection.

First of all, the autoregressive model in (4) is reformulated as

$$\begin{aligned} y_t = & \sum_{s_j \in \Omega_p} \left(\beta_{j,0} + \sum_{\tau=1}^{l} \beta_{j,\tau} x_{j,t-\tau} \right) \\ & + \sum_{s_j \in \Omega_a} \left(\beta_{j,0} + \sum_{\tau=1}^{l} \beta_{j,\tau} x_{j,t-\tau} \right) + \varepsilon_t \end{aligned} \tag{7}$$

so that the original intercept terms is shared among all sites, which will simplify some of the derivations below. The dimension of $\boldsymbol{\beta}$ hence increases to $(l+1)(m_a + m_p)$.

The Lasso estimator is defined as

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{y} - \mathbf{A}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 \tag{8}$$

where $\mathbf{y}$ is a vector containing all response variable values over the time indices in Γ, while $\mathbf{A}$ is the design matrix, where its ith line is defined by $\mathbf{x}_i$ (as in (5)). λ serves as a weight to control the trade off between LS fitting and shrinkage of $\boldsymbol{\beta}$.

This problem could be solved in a centralized manner, by gathering data for all explanatory variables and lags. This would translate to a necessary data exchange between the central agent and the contracted agents. This is what we aim to avoid though, since contracted agents may be happy to help with distributed learning, but only under the condition of never sharing data. In view of that basic, but crucial, characteristics of our learning problem, we proceed by decomposing the global learning problem in (8), so as to obtain a number of distributed learning problems which could be shared among contracted agents. For that, we follow the approach referred to as the alternating direction method of multipliers (ADMM, [7]), which is a widely used decomposition approach based on Lagrangian relaxation that allows to split a learning problem among features. Here in practice, this would translate to decomposing the learning problem among contracted agents. Following the reasoning in [7], let us first relax the learning problem in (8) such that it becomes

$$\min \quad \frac{1}{2}\|\mathbf{y}-\mathbf{A}\boldsymbol{\beta}\|_2^2+\lambda\|\mathbf{z}\|_1 \tag{9}$$

$$\text{s.t.} \quad \boldsymbol{\beta}-\mathbf{z}=0 \tag{10}$$

Following the ADMM approach, the learning problem is then decomposed among the contracted agents, by setting

$$\boldsymbol{\beta}=[\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\ldots,\boldsymbol{\beta}_{m_a+m_p}] \tag{11}$$

$$\mathbf{A}=[\mathbf{A}_1\,\mathbf{A}_2\,\ldots\,\mathbf{A}_{m_a+m_p}] \tag{12}$$

so that each contracted agent j is focused on estimating its own share $\boldsymbol{\beta}_j$ of the parameter vector based on his local data in the form of the design matrix $\mathbf{A}_j$.

ADMM is an iterative approach where one alternates between the contracted agents solving their learning sub-problems, the solution of which is transmitted back to the central agent, who solves a master learning problem. At iteration k, this writes

$$\boldsymbol{\beta}_j^k=\underset{\boldsymbol{\beta}_j}{\operatorname{argmin}}\left(\|\mathbf{A}_j\boldsymbol{\beta}_j-\mathbf{y}_j^{k-1}\|_2^2+\frac{2\lambda}{\rho}\|\boldsymbol{\beta}_j\|_1\right) \tag{13}$$

$$\overline{\mathbf{z}}^k=\frac{1}{(l+1)(m_a+m_p)+\rho}\left(\mathbf{y}+\overline{\mathbf{A}\boldsymbol{\beta}}^k\rho u^{k-1}\right) \tag{14}$$

$$\mathbf{u}^k=\rho u^{k-1}+\overline{\mathbf{A}\boldsymbol{\beta}}^k-\overline{\mathbf{z}}^k \tag{15}$$

where

$$\mathbf{y}_j^{k-1}=\mathbf{A}_j\boldsymbol{\beta}_j^{k-1}-\overline{\mathbf{A}\boldsymbol{\beta}}^{k-1}+\overline{\mathbf{z}}^{k-1}-\mathbf{u}^{k-1} \tag{16}$$

$$\overline{\mathbf{A}\boldsymbol{\beta}}^k=\sum_{j=1}^{m_a+m_p}\mathbf{A}_j\boldsymbol{\beta}_j \tag{17}$$

In the above algorithm, the updates in (13) are local updates for the partial local estimates, while the z-update in (14) is for the central agent to iterate on its full set of parameters, to obtain $\boldsymbol{\beta}$, eventually.

A number of convergence criteria for this algorithm are proposed in [7]. In a straightforward manner, one may monitor the norm of the increment $\boldsymbol{\beta}^k-\boldsymbol{\beta}^{k-1}$ and stop when this norm gets small, say, 10^{-3}. Similarly, a maximum number of iterations should be defined, e.g., such that $k\leq 500$. The final parameter vector estimate $\hat{\boldsymbol{\beta}}$ is then obtained by assembling the local estimates $\boldsymbol{\beta}_j^k$ at convergence (as in (11)). The ADMM algorithm may be initialized by setting all relevant variables to 0. Besides the structure of the underlying autoregressive model originally chosen, this algorithm has two parameters λ and ρ, which may be selected through cross-validation. ρ is an augmented Lagrangian parameter that controls the step size in the iterative optimization performed by the ADMM approach.

This ADMM approach permits to coordinate distributed learning by this iterative exchange between the master problem of the central agent and the sub-problems of the contracted agents. Though, the contracted agents still need to locally solve the Lasso estimation problem (13) at each iteration k. There, we propose to use an efficient and simple approach nicknamed "shooting" [9]. Shooting is also an iterative procedure, which combines simplicity in terms of the iterative procedure it implements and suitable convergence to the Lasso estimate, in comparison to alternative approaches, e.g., quadrative programming variants, interior-point methods, etc. Considering we want to compute $\boldsymbol{\beta}_j^k$, the shooting algorithm is to be initialized with the ridge regression estimator corresponding to the Lasso estimator in (13). It is given by

$$\tilde{\boldsymbol{\beta}}_j^k=\left(\mathbf{A}_j^\top\mathbf{A}_j+\frac{2\lambda}{\rho}\mathbf{I}\right)^{-1}\mathbf{A}_j^\top\mathbf{y}_j^{k-1} \tag{18}$$

Subsequently, starting from this estimate, one loops over its element $i=1,\ldots,l+1$, by first calculating

$$S_i=-\mathbf{A}_{j,i}^\top\mathbf{y}_j^{k-1}+\sum_{n\neq i}\mathbf{A}_{j,i}^\top\mathbf{A}_{j,n}\beta_i \tag{19}$$

where $\mathbf{A}_{j,i}^\top$ denotes the ith column of the design matrix $\mathbf{A}_j^\top$, and then update the variable β_i such that

$$\beta_i=\frac{1}{\mathbf{A}_{j,i}^\top\mathbf{A}_{j,i}}\left(\frac{2\lambda}{\rho}-S_i\right),\quad S_i>\frac{2\lambda}{\rho} \tag{20}$$

$$\beta_i=-\frac{1}{\mathbf{A}_{j,i}^\top\mathbf{A}_{j,i}}\left(\frac{2\lambda}{\rho}-S_i\right),\quad S_i<-\frac{2\lambda}{\rho} \tag{21}$$

$$\beta_i=0,\quad |S_i|\leq\frac{2\lambda}{\rho} \tag{22}$$

This looping over the elements is repeated until a convergence criterion is met (e.g., related to increment norm, or maximum number of iterations). The local estimate $\boldsymbol{\beta}_j^k$ for the ADMM algorithm at iteration k is then given by β_i, $i=1,\ldots,l+1$.

IV. Application Results

The distributed learning approach presented in the above is applied and analyzed here based on 2 datasets in France and Australia. Results are given for all different application cases discussed in the above, i.e., for the case of single wind farm operators getting support from other wind farms, for the case of portfolios of wind farms that contract others to improve their forecasts, as well as the case of a system operator that would distribute his learning problem to all wind farms in his system. Only one-step ahead forecasts are considered here

for illustration, also since emphasis is placed on models for very-short-term forecasting.

A. Australian and French Datasets

The Australian dataset was originally constructed based on data made publicly available by the Australian Electricity Market Operator (AEMO). The temporal resolution of the data is of 5 minutes, for a set of 22 wind farms in South Australia, over a period of nearly 2 years. All power measurements are normalized by the nominal capacity of the wind farm they relate to. No meteorological measurement (e.g., wind speed and direction) is available. A link to this data is given in [4]. The temporal resolution of the data was coarsened to time steps of 30 minutes, by averaging the 6 5-minute values in each and every 30-minute interval. This allows to have a lighter dataset to run many test cases examples to be reported here.

A similar dataset from France is used. Power measurements with a temporal resolution of 10 minutes were originally collected at 187 wind farms over a period of 3 years and 5 months. For confidentiality reasons, locations and characteristics of the wind farms cannot be communicated here. Power measurements are also normalized, while no meteorological measurement is associated to this power data. After quality checking of these power observations, and coarsening to a hourly resolution (by an averaging procedure similar to the Australian data case), one is left with power measurements for 85 wind farms over the whole period.

B. Evaluation Framework

As is commonly done in wind power forecasting, we split our datasets between a learning/cross-validation set Γ_l and an evaluation set Γ_e. The former dataset is used to make decisions on model structure, the penalization λ for the Lasso estimator, the step size ρ for the distributed learning approach, etc. Cross-validation is employed to decide upon these parameters while aiming to maximize the generalization ability of the model for forecasting. Over the evaluation set Γ_e, the models are readily applied to predict power generation as if in operational conditions.

The lead score we use to compare forecasts is the Root Mean Square Error (RMSE). Based on the forecast error

$$\epsilon_{t+1|1} = y_{t+1} - \hat{y}_{t+1|t}, \quad \forall t \in \Gamma_e \tag{23}$$

calculated for each time instance in the evaluation set, the RMSE is readily given by

$$\text{RMSE} = \left(\frac{1}{T_e} \sum_{t \in \Gamma_e} \epsilon_{t+1|1}^2 \right)^{\frac{1}{2}} \tag{24}$$

where T_e is the number of forecast-observation pairs in the evaluation set Γ_e. RMSE is one of the relevant criteria to evaluate point forecasts of wind power generation, especially since forecasts are defined as conditional expectation of wind power generation at time $t+1$ given information, model and parameters estimated at time t. For an extensive overview of approaches to wind power forecast verification, the reader is referred to [10].

Finally, as a basis for comparison, a number of common benchmark models are used. Since the premise of our work is that the central agent only have access to limited amount of data, relevant benchmark models include persistence and autoregressive (AR) models. For the former, no decision upon model structure or estimation is to be carried out, while this is not the case for the latter. A Lasso estimator is also used here for the AR models, hence allowing for automatic selection of relevant lags.

C. Illustration for a Single Wind Farm

To first illustrate the application and interest of our distributed learning framework, we place emphasis on the case of a single wind farm operator, who aims at predicting power generation at his site of interest, though potentially profiting of potentialoffsite information to improve his forecasts. For that illustration, we consider wind farm 8 in the Australian dataset, following the wind farm numbering proposed in [4]. Even though the wind farm operator may restrict itself to contracting the closest wind farms for distributed learning, since being of most relevance anyway, we assume here that contracts are made with the other 21 wind farms. In practice with this set-up, the actual model to be estimated is an ARX model (AR with exogenous input) since having an AR part as well as offsite information. A number of maximum 5 lags is used (plus intercept term), while the Lasso estimator allows to shrink the model and only select those lags that are relevant. 2.000 times steps (hence covering 1.000 hours) are used for estimating the models, while the following 10.000 are employed for genuine forecast verification. The forecasting results are collated in Table I.

TABLE I: Comparative results for distributed learning (ARX model), as well as persistence and AR benchmarks, at an Australian wind farm (wind farm no. 8) for 30-min ahead forecasting.

	Persistence	AR	ARX (dist. learning)
RMSE [% nom. capacity]	3.60	3.57	3.52
Improvement [%]	-	0.8	2.2

For this wind farm (and for this whole dataset in general, as shown in [4]), it may be very difficult to outperform persistence with such noisy response data while considering very short lead times. In the present example, an AR model with parameters obtained through application of a Lasso-type estimator allows selecting the 2 closest lags as input variables, while yielding a RMSE improvement of 1%. In comparison, applying a distributed learning approach, hence aiming to improve forecasts by capturing space-time dependencies while never sharing data at the various sites, the RMSE is further improved (reduction of 2.2%). Out of the 22×6 variables (5 lags and the intercept term) that could be selected as input, only 3 were selected as relevant explanatory variables. This approach therefore allows obtaining very sparse models combining local and offsite information. It is to be noted that owing to the nonstationary nature of wind power dynamics, it may actually be that relevant explanatory variables change

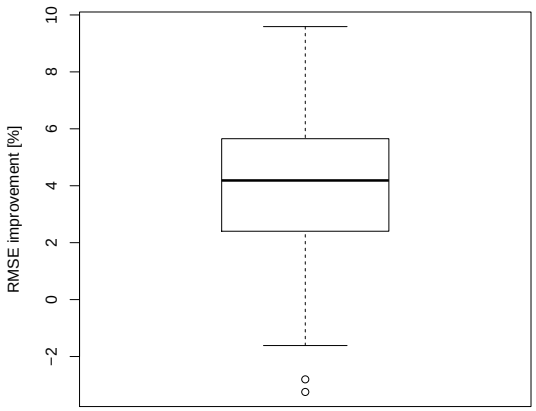

Fig. 2: Box-plot representing the distribution of RMSE improvements (in %) for all 85 wind farms, when comparing results obtained with local information only (AR model) and with offsite information through distributed learning (hence, with ARX models).

with time, being different lags and corresponding to different locations.

D. Results for Wind Farms and Portfolios

Let us now consider a more complete setup with the French dataset. With this dataset, we aim at looking at a broader range of relevant application of this distributed learning approach. First of all, it allows us to verify how general benefits from distributed learning may be, for a large number of wind farms. For that purpose, we repeat an experiment similar to that in the above, by looking at the case of each and every wind farms individually, comparing cases where they use local data only and where they would use a distributed learning approach. Forecasts are all issued for a 1-hour lead time. A period is randomly chosen within the dataset, with training over 5000 consecutive hours, and then forecast evaluation over the following 10.000 hours. Models are built with up to 3 lags and an intercept term for each and every wind farm. In the distributed learning case, this translates to models with up to 85×4 non-zeros parameters. Results are presented in the form of a box-plot in Figure 2.

Improvements in RMSE values exist for nearly all wind farms, with a 4.08% average and a range between -3.2 and 9.6%. In all cases, improvements over persistence are highly significant, one order of magnitude larger than for the Australian dataset. Here, we believe that such negative values in RMSE improvements, which are there for a minority of wind farms, may be explained by the nonstationarity of the wind power dynamics, since the model is estimated on a certain period in the past, and then assumed to be valid for the following 10.000 hours. Remedies to that issue would include batch estimation on sliding windows, or online distributed learning approach, hence with recursive and adaptive estimation of the parameters in a distributed learning environment. The latter solution should be preferred, since batch re-estimation would be heavier from a computational point of view while not necessary yielding better forecasting results. Through the use of a Lasso-type estimator, very few of the possible 370 model parameters ended up to be non-zeros, and this for all wind farms.

We now look at the more general case of a portfolio manager who aims at improving forecasts for his whole portfolio, possibly profiting of distributed learning to gain from offsite without actually seeing the data from these other sites. For illustration, we use a set-up similar to the case above (time periods, model structures, etc.), though having assembled randomly a portfolio composed by 8 wind farms from this French datasets. Results are gathered in Table II. Profiting of smoothing effects, forecast errors are lower in general. In addition, improvements over persistence are more significant. The benefits from distributed learning appear to be more significant too, most likely since for a geographically distributed portfolio they may be even more to learn from all offsite locations.

TABLE II: Comparative results for distributed learning (ARX model), as well as persistence and AR benchmarks, for a portfolio of 8 wind farms of the French dataset (randomly chosen) for 1-hour ahead forecasting.

	Persistence	AR	ARX (dist. learning)
RMSE [% nom. capacity]	3.99	3.67	3.38
Improvement [%]	-	8.2	15.3

Finally considering the case of the system operator, we envisage here the situation where the forecasting problem for the aggregate power of the 85 wind farms is fully distributed among all sites, meaning that this system operator does not even have to know about the individual production data at each and every site. The set-up is here again similar to the previous forecasting experiments. Results are summarized in Table III. RMSE values are even lower here, while there is not really relevant offsite information when considering the aggregate of all wind farms. However, one sees that there is still a little benefit from considering all individual sites, with a few additional percent decrease in RMSE values.

TABLE III: Comparative results for distributed learning (ARX model), as well as persistence and AR benchmarks, for the aggregate of all 85 French wind farms for 1-hour ahead forecasting.

	Persistence	AR	ARX (dist. learning)
RMSE [% nom. capacity]	2.88	2.10	2.05
Improvement [%]	-	27.1	28.8

V. Conclusions

We have proposed and studied a distributed learning approach for wind power forecasting, which offers a flexible framework for estimating models that may allow to account for information at a large number of sites without having to exchange actual production data at these sites. There is a wide range of applications for forecast improvement, for single sites

to portfolios wind farm operators and portfolio manager, and to whole aggregate of wind farms for system operators.

This distributed learning approach may be extended and generalized in many ways. First of all, one may consider using additional input information, e.g., weather forecasts, if looking at further lead times. Then, from a more methodological point of view, other estimators than Lasso-type ones may be implemented and studied. Online learning approaches would allow to lighten the computational burden while adapting to the nonstationary dynamic behavior of wind power generation. Eventually, distributed learning can be used to generate probabilistic forecasts, instead of the deterministic ones considered here.

On a side note, this concept of distributed learning allows envisaging the development of a market place for data sharing that would benefit forecast quality while rewarding monetarily those ready to contribute to forecast improvement. Pricing information sharing may be challenging, though possibly naturally formulated in a game-theoretic framework.

ACKNOWLEDGMENT

The authors would like to thank AEMO and ENEDIS for providing the data used for the test case applications in this paper. Besides, the authors acknowledge EDF R&D for partly supporting this work through the project "HD-RESforecasts". The authors are finally grateful to a number of reviewers for their comments and suggestions on an earlier version of that manuscript.

REFERENCES

[1] A.M. Foley, P.G. Leahy, A. Marvugliac, and E.J. McKeogh, "Current methods and advances in forecasting of wind power generation," *Renew. Energ.*, vol. 37, no. 1, pp. 1-8, 2012.

[2] R.C. Inman, H. Pedro, and C.F.M. Coimbra, "Solar forecasting methods for renewable energy integration," *Prog. Energ. Combust. Sci.*, vol. 39, no. 6, pp. 535-576, 2013.

[3] P. Pinson, "Wind energy: Forecasting challenges for its optimal management," *Statist. Science*, vol. 28, no. 4, pp. 564-585, 2013.

[4] J. Dowell, and P. Pinson, "Very-short-term wind power probabilistic forecasts by sparse vector autoregression," *IEEE T. Smart Grid*, vol. 7, no. 2, pp. 763-770, 2016.

[5] R.J. Bessa, A. Trindade, A. Monteiro, C.S.P. Silva, and V. Miranda, "Solar power forecasting in smart grids using distributed information," in *Proc. of the 18th Power Systems Computation Conference (PSCC 2014)*, Wroclaw, Poland, Aug. 2014.

[6] L. Cavalcante, R.J. Bessa, M. Reis, and J. Dowell, "Sparse structures for very short-term wind power forecasting", working paper, 2016.

[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical Learning via the Alternating Direction Method of Multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1-122, 2011.

[8] P. Pinson, "Very short-term probabilistic forecasting of wind power with generalized logit-Normal distributions," *J. Royal Stat. Soc. C*, vol. 61, no. 4, pp. 555-576, 2012.

[9] W.J. Fu, "Penalized regressions: The bridge versus the lasso," *J. Comput. Graph. Statist.*, vol. 7, no. 3, pp. 397-416, 1998.

[10] H. Madsen, P. Pinson, G. Kariniotakis, H.Aa. Nielsen, and T.S. Nielsen, "Standardizing the performance evaluatiof short-term wind power prediction models," *Wind Eng.*, vol. 29, no. 6, pp. 475-489, 2005.