Galván, I.M., Valls, J.M., Cervantes, A., Aler R. (2017). Multi-objective evolutionary optimization of prediction intervals for solar energy forecasting with neural networks. Information Sciences, (418-419), pp. 363-382.

DOI: 10.1016/j.ins.2017.08.039

Noname manuscript No.
(will be inserted by the editor)

Multi-objective optimization of prediction intervals for solar energy forecasting with neural networks

Ricardo Aler ⋆ **· José M. Valls ·**
Alejandro Cervantes · Inés M. Galván

the date of receipt and acceptance should be inserted later

Abstract In the context of forecasting for renewable energy, it is common to produce point forecasts but it is also important to have information about the uncertainty of the forecast. To this extent, instead of providing a single measure for the prediction, lower and upper bound for the expected value for the solar radiation are used (prediction interval). This estimation of optimal prediction intervals requires simultaneous optimization of two objective measures: on one hand, it is important that the coverage probability of the interval is as close as possible to a given target value. On the other, in order to bound uncertainty, intervals must be narrow; this means that there is a trade-off between both objectives, as narrow intervals reduce the coverage probability for those solutions, as the actual value of solar radiation is more likely to fall outside the predicted margins. In this work we propose a multi-objective evolutionary approach that is able to optimize both goals simultaneously. The proposal uses neural networks to create complex non-linear models whose outputs are the upper and lower limits of the prediction intervals. Results have been compared with a single-objective optimization of similar neural network architectures and a baseline algorithm (quantile estimation with gradient boosting). Results show that the neural network is able to provide accurate results. Also, the multi-objective approach is able to obtain the best results and has also the advantage that a single run is able to obtain prediction intervals for any target coverage probability.

Keywords Estimation of Prediction Intervals · Neural Networks · Multi-objective Particle Swarm Optimization · Solar Energy Forecasting

⋆ Corresponding author: Ricardo Aler, Tel.: +34-91-6249418, Fax: +34-91-6249430, E-mail: aler@inf.uc3m.es

Ricardo Aler · José M. Valls · Alejandro Cervantes · Inés M. Galván
Universidad Carlos III de Madrid
Avda. Universidad 30
28911 Leganés, Spain
E-mail: {aler, jvalls, acervant, igalvan }@inf.uc3m.es

1 Introduction

The importance of renewable energy in the electric energy production mix has increased in recent years. But because of its inherent variability, a crucial issue for integrating renewable sources in the energy mix is to improve the forecast reliability of the sources (solar, wind, ...). Many research works can be found in the literature that achieve this aim by means of statistical or machine learning methods (Mellit, 2008; Costa et al, 2008; Mellit and Pavan, 2010; Pedro and Coimbra, 2012; Marquez and Coimbra, 2011; Marquez et al, 2013; Alonso et al, 2015; Linares-Rodriguez et al, 2015; Gala et al, 2016; Okumus and Dinler , 2016). In most cases, the models are trained to produce point forecasts or average predictions for a period of time. However, in the context of renewable energy it is also important, not only to have accurate forecasts, but to estimate their uncertainty (Pinson, 2013). A common way of estimating uncertainty is probabilistic forecasting (Zhang et al, 2014). Probabilistic forecasting can be carried out by estimating quantiles, probability density functions (PDFs), and prediction intervals. Quantiles (such as the median or the first percentil) can be estimated by means of quantile regression techniques (Nielsen et al, 2006; Bremnes, 2004) and also machine learning methods such as radial basis neural networks (Sideratos and Hatziargyriou, 2012). PDFs can be approximated by means of kernel density estimators (Juban et al, 2007). In the case of prediction intervals (PIs) the aim to is to estimate lower and upper limits that bound the forecast for a given confidence level.

This paper is focussed on probabilistic forecasting using PIs. In (Khosravi et al, 2011) four traditionally used methods to estimate PIs (delta, bayesian, bootstrap, and mean-variance) are analyzed, but argues that none of them aim to optimize both main defining features of PIs, namely coverage and interval width, in addition to being computationally costly. Instead, an approach to estimate directly the lower and upper bounds of the PIs is proposed and shown to be equal or superior to the other methods in different domains. This direct approach optimizes artificial neural networks (ANN), whose outputs are the lower and upper bounds of the PI, by means of an evolutionary computation technique known as simulated annealing. In (Khosravi and Nahavandi, 2013), this direct approach is applied to estimating PIs for wind energy production. Since then, the direct estimation of wind energy PI bounds, by optimizing machine learning models with evolutionary methods, has been applied in different ways. For instance, (Kavousi-Fard et al, 2014) uses ANNs, fuzzy logic and particle swarm optimization (PSO) in order to optimize different criteria that evaluate the quality of PIs. A combination of type-2 fuzzy systems and PSO is described in (Marin et al , 2016). In (Wan et al, 2016), PSO optimizes an extreme learning machine (ELM) (a type of ANN) whose two outputs represent the PI. In the latter work, the interval does not depend on the meteorological situation. This issue has been addressed in (Pinson and Kariniotakis, 2010), where intervals are conditioned to some meteorological variables by means of fuzzy logic, but evolutionary computation was not used in this case. In (Afshari-Igder et al, 2016) PIs are computed taking into account

two sources of variability: model and noise, ELMs and the bootstrap method is used for the former and the improved krill herd optimization algorithm (an evolutionary method) for the latter.

All previous works refer to wind forecasting. There are fewer works for solar energy probabilistic forecasting. For instance, in (Bacher et al, 2009), autoregressive models are used for point forecasting and quantile regression to estimate uncertainty. In (Bracale et al, 2013) the probability distribution of the energy generated by a photovoltaic plant is approximated with a bayesian autoregression approach. (Chu et al, 2015) proposes a hybrid model using support vector machines (SVM) to classify situations into high and low DNI (direct normal irradiance) variation, and then ANNs to approximate the variance of the PI. In (Alessandrini et al, 2015), analog ensembles (a technique related to nearest neighbor methods), is used for short term probabilistic solar power forecast. With respect to PIs, in a recent work, the direct approach already discussed for wind energy is also applied to solar energy forecasting (Chai et al, 2015). In particular, a granular neural network is trained using PSO to approximate the upper and lower PI bounds.

Based on the successful application of the direct estimation of PI bounds discussed in the previous literature review, the present work takes this technique as a foundation and extends it using multi-objective (MO) evolutionary optimization methods. This work will be empirically tested on solar forecasting, where less research has been carried out. Following other works in the literature, a multi-layer perceptron (MLP) with two outputs are used to determine the lower and upper interval bounds. Given that the desired outputs (the interval limits) are not known, the standard backpropagation learning algorithm cannot be used to train the network. In that situation, previous research has shown that evolutionary computation can be used to optimize the MLP and therefore, the returned PI bounds. The two most important criteria to evaluate PIs are the width of the interval (to be minimized) and its coverage probability (to be maximized). While the standard direct estimation of PIs uses those two conflicting goals to evaluate PIs, typically they are optimized using single-objective (SO) methods. SO techniques address multi-criteria problems by aggregating both goals into a single value using a reasonable tradeoff between them. This tradeoff is controlled by means of a parameter or weight, and different weights will produce solutions of different qualities. However, the weight has to be set by the user prior to the optimization process and might not return a PI with the appropriate coverage or width, requiring optimization to be repeated with a different weight. In addition, the single solution returned by SO does not allow the user to observe the trade-offs between the two goals. For instance, the user cannot determine how much the interval width would have to be increased in order to have a larger coverage.

The main motivation of the present work is to address these issues by using multi-objective (MO) evolutionary techniques (Deb , 2001; Talbi, 2009; Babalk , 2017), more concretely, multi-objective particle swarm optimization (MOPSO) (Coello et al, 2004). Hence, in this work MLPs with two outputs (the limits of the intervals) are trained using MOPSO. Instead of returning

a single solution, MO techniques provide a set of them, called the Pareto front. This set contains the solutions with the optimal trade-offs between the different objectives to be optimized. In the proposed approach, each solution in the final Pareto front is a trained MLP, each one generating PIs with different trade-offs between interval coverage and width. From this front, the user can select the most appropriate solution to his/her needs, in terms of the desired coverage or width. But differently to the SO approach, the user does not need fix his/her preferences before the optimization process, because the MO generates the whole Pareto front, from which selection can take place after the optimization run. One of the most important features of evolutionary MO techniques is that they include mechanisms to encourage the uniform distribution of solutions along the front. This makes it likely that solutions with the required properties will be present in the front. Additionally, having the whole Pareto front available, and with all regions in the front covered, allows the user to explore the effect of small changes to his/her preferences. For instance, it could be determined the cost in terms of interval width increase of having a slightly larger coverage.

The rest of the paper is organized as follows. In Section 2 the problem of estimating PIs is defined, including the functions to measure the quality of an interval. Section 3 presents briefly the PSO and MOPSO algorithms. The multi-objective approach proposed in this work is described in Section 4. Section 5 describes the data used for the experiments. The experimental results obtained by the proposed approach are shown in Section 6. This section also includes a comparison with a single-objective approach using PSO and a baseline in this field (quantile estimation by gradient boosting). Finally, Section 7 draws the main conclusion of this work.

2 Estimation of Prediction Intervals

Let $M = \{(X_i, t_i)_{i=0 \cdots N}\}$ be a set of observations, where X_i is a vector with the input variables and t_i is the observed output variable. In the context of renewable energy, X_i are the observed values of some meteorological variables at moment i, and t_i can be the solar or wind energy generated at that time. A solution to the estimation of intervals is a sequence of PIs, $PI_i = [Low_i, Upp_i)$, that is defined for the whole set of observations $X_i, i \in [0 \cdots N]$. Thus, given a new observation X_k $k > N$, a corresponding PI, $PI_k = [Low_k, Upp_k)$, should be provided. In summary, for every possible input X_i, the intention is to compute its optimal PI, $PI(X_i) = [Low(X_i), Upp(X_i))$, that contains the true value t_i of the output variable.

Note that in this scenario, the point estimate for solar radiation is not the objective of the prediction. Quality of a solution is assessed by the number of measures that lie in the predicted PI. This is the reason why narrow PIs are desirable.

The quality of a solution can be evaluated using two conflicting measures: **reliability** and **width**. The **reliability** of a PI is the expectation that for any

future observation X_k, t_k will lay inside the interval PI_k. **Reliability** is measured using the prediction interval coverage probability ($PICP$), calculated using Eq. 1 over the available data.

$$PICP = \frac{1}{N}\sum_{i=0}^{N}\chi_{PI_i}(X_i) \tag{1}$$

where $\chi_{PI_i}(X_i)$ is the indicator function for interval PI_i, that takes value 1 if $t_i \in PI_i$ and value 0 otherwise.

Obviously, **reliability** can be made larger by just increasing the **width** of the PI, but then we will be less certain about the actual output value. Therefore, it can be considered that the **width** of the interval measures the degree of uncertainty around a prediction. Given that different observations have different PIs, the average interval width (AIW) can be calculated using Eq. 2 over the available data.

$$AIW = \frac{1}{N}\sum_{i=0}^{N}(Upp_i - Low_i) \tag{2}$$

where Upp_i and Low_i are the upper and lower bounds of the interval, respectively. In other works, an interval score derived from Upp_i and Low_i, is used instead of AIW. The interval score rewards narrow PI and gives penalty if the target does not lie within the estimated PI (Wan et al, 2014). However, this measure evaluates both width and the actual output belonging to the interval, but the latter is closely related to **reliability**. In this article, we have preferred to have two independent measures, as will be explained in the next section.

It is important to note that most practical problems involve the estimation of PIs that will be able to achieve an specified **reliability** level. This means that a nominal value will be defined as a parameter and the optimization methods will provide the best solution that meets the criterion of being close to this nominal level. This value is usually called prediction interval nominal confidence ($PINC$) and in order to meet that restriction, solutions must be found where $PICP \geq PINC$.

In summary, our interest in this article is, given some available data M and a target $PINC$, to find models Upp and Low that for every input X return the PI $[Low(X), Upp(X))$, such that:

- $PICP \geq PINC$
- AIW is as small as possible.

3 Single and Multi-objective Particle Swarm Optimization

Particle Swarm Optimization (PSO) (Kennedy et al, 2001) is an iterative stochastic search method that is designed for finding the optimum value for a fitness function that has a number of parameters that take values either in

a discrete or in a real-valued parameter space. However PSO is mostly used for real-value parameter optimization. PSO belongs to the family of nature-inspired metaheuristic algorithms, which includes genetic algorithms, evolution strategies, harmony search, or ant colony optimization, among others, and they have shown a good performance in engineering and mathematical optimization problems (Arqub and Abo-Hammour , 2014; Mousavi and Alfi , 2015; Arab and Alfi , 2015; Ameli et al , 2016; Pahnehkolaei et al , 2017).

In PSO, search is conducted by a population (swarm) of solutions (particles). Particles are defined by their position vectors $\mathbf{X}_i = \{x_{i0}, x_{i1}, \cdots, x_{im}\}$ where each x_{ij} is the value of one of the parameters of the function f to be optimized, called objective function.

There are several versions of PSO adapted for multi-objective optimization. One of the most popular versions is MOPSO (Coello et al, 2004), that uses the archiving strategy of PAES (Knowles and Corne, 2000). In this work we use a variant of this algorithm that introduced crowding distance to ensure maximum coverage of the Pareto front (Raquel and Naval , 2005).

In MOPSO, fitness is a vector $\mathbf{F} = \{f_0, f_1, \cdots, f_n\}$ with one value f_i for each objective function. Each particle in MOPSO is a solution to the optimization problem. The algorithm uses an external repository that contains the Pareto front of non-dominated solutions found during its execution and returns the repository when reaches its termination criterion.

Both in PSO and MOPSO, the particle's movement is guided by two influences. One is the best position of the particle at the moment. This ensures that each particle performs a local search of its environment. The second influence is a neighbor solution selected from the rest of the swarm, that is chosen in different ways in PSO and MPSO:

- In PSO, particles are statically "connected" to a set of other particles, called the particle's neighborhood. The best particle in that neighborhood is used to influence the original particle's movement.
- In MOPSO, the algorithm stores non-dominated solutions in an external repository. Each particle is influenced by one of the solutions already present in this repository, that is, using a random procedure that favors solutions in areas of the repository that are sparsely populated to the moment.

The MOPSO algorithm tries to generate non-dominated solutions with due diversity, that is, solutions must be spread along a Pareto Front, and not concentrated in a specific section in objective space. Diversity arises from the mechanism that each particle uses to select the leader solution that guides its movement on a given iteration.

The variant of MOPSO used in this work selects leaders for the particles in the population using a metric called crowding distance (Raquel and Naval , 2005). For every solution in the archive, its crowding distance is calculated comparing the position in objective space with the positions of the closest solutions in the archive. Non-dominated solutions with maximum distance to their neighbors in objective space are therefore considered more isolated and

used as leaders for each iteration of the algorithm. These values are updated whenever new solutions are inserted in the MOPSO archive.

4 Multi-objective approach to estimate Prediction Intervals

In this article, the models that estimate the upper Upp and lower Low bounds of a PI take the shape of a MLP with two outputs (Upp and Low). The inputs to the MLP are meteorological variables available near the locations of solar stations. The network is a three-layered perceptron with a single layer of hidden units. In this context, the target outputs (upper and lower bounds) for the MLP are not directly available and therefore it cannot be trained using the standard back-propagation algorithm. Instead, an optimization method that does not depend on gradients, such as PSO, can be used for that purpose. PSO is also a good option for real-value optimization, which is the problem addressed in this article (optimizing the weights of MLPs). As explained in Section 2, models must produce intervals that are both close to the nominal probability coverage ($PINC$) and narrow.

It is straightforward to formulate an optimization problem where a weighted aggregation of both goals is used as the function to be optimized (exact details of a possible formulation can be found in Subsection 4.2). However, if this approach is used, the user has to commit on the relative importance (i.e. the weight) of each goal before the optimization process starts. The final result will depend on the chosen weight, and if the final PI does not conform to the user needs, the optimization process will have to be re-run with a different weight.

In our proposal, the problem is addressed within a multi-objective formulation, where both goals are optimized simultaneously using MOPSO. Details of the function to be optimized can be found in Subsection 4.2. After a single run of MOPSO, a Pareto front of non-dominated solutions is generated, which means that, for each value of $PINC$, the model that obtains the narrowest PI is obtained. This formulation removes the need to define the weight that determines the tradeoff between the two goals, because optimal solutions for all possible tradeoffs are obtained in a single run. In a real-world scenario, this would have the added advantage that the prediction module could be adjusted instantly to a different level of $PINC$ if desired, without running the optimization algorithm again.

When using evolutionary algorithms (like PSO and MOPSO), both the decision and the objective spaces have to be defined:

- Decision space: in our case it is the set of weights that define the MLP models that compute PIs, which in evolutionary algorithms must be encoded in the chromosome. Next Subsection 4.1 explains this.
- Objective space: for this problem, it is defined by two goals: closeness to the target $PINC$ and interval width. In evolutionary algorithms, goals are computed by means of fitness functions, which are explained in Subsection 4.2.

4.1 The decision space: encoding MLP weights

In this work, single-hidden-layered MLPs are used to represent the models. Both the single hidden layer and the output layer have sigmoidal units and their activation functions are given by Eq. 3:

$$s(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

where $s(x)$ is a real-valued function associated to the sigmoidal unit, and x corresponds to the total input to that neuron.

The output layer has two sigmoidal units whose outputs (y_1 and y_2), represent the upper and lower bounds of the interval, respectively. Figure 1 displays the architecture of such ANN. Thus, given an input vector $\mathbf{x} \in \mathbb{R}^n$, the output or activation of the i_{th} hidden neuron, a_i, is given by

$$a_i = s(\mathbf{w}_i^t \cdot \mathbf{x} + b_i) \tag{4}$$

where $\mathbf{w}_i = (w_{1i}, w_{2i}, \ldots, w_{ni})$ is the vector of weights connecting the n inputs and the i^{th} hidden neuron and b_i is the threshold or bias corresponding to the i^{th} hidden neuron.

The output of the first output neuron, y_1, when the hidden layer has m units, is given by:

$$y_1 = s(\sum_{i=1}^{m} h_{i1} \cdot a_i + b_1^o) \tag{5}$$

where h_{i1} is the weight connecting the i^{th} hidden neuron and the first output neuron, a_i is the activation of the i^{th} hidden neuron, given by Eq. 4, and b_1^o is the bias corresponding to the first output neuron.

Similarly, the output of the second neuron y_2 is given by:

$$y_2 = s(\sum_{i=1}^{m} h_{i2} \cdot a_i + b_2^o) \tag{6}$$

For m hidden neurons in the MLP, the number of weights for n inputs and 2 outputs is given by Eq. 7:

$$D = \overbrace{n \times m}^{w_{ji}} + \overbrace{m}^{b_i} + \overbrace{m \times 2}^{h_{ik}} + \overbrace{2}^{b_k^o} \tag{7}$$

where $k = \{1, 2\}$ corresponds to the output neuron.

Both PSO and MOPSO optimize the values of the D weights of the MLP by encoding the whole set of weights and bias as coordinates for each of the particles, and performing a search in $\mathbb{R}^D$ for the values that provide the best results.

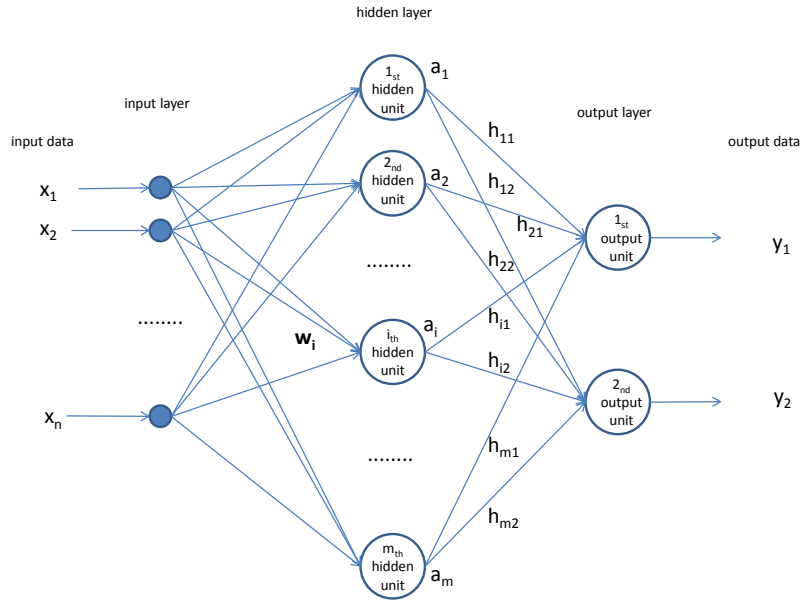

Fig. 1 Multilayer perceptron with n inputs and m hidden units used to estimate the upper and lower bounds of the PI

4.2 The objective space: fitness functions for PSO and MOPSO

As explained in section 2, the quality of a PI depends on both coverage probability ($PICP$) and interval width (AIW). For a single-objective formulation, both goals have to be aggregated by means of a weight which measures the tradeoff between the two goals. In particular, the fitness function for PSO, which must be minimized, is given by Eq. 8:

$$F_{PSO}(PINC) = a_1 * AIW + (1 - a_1) * \max\{PINC - PICP, 0\} \quad (8)$$

The first goal computes AIW (see 2) and the second goal measures the closeness of $PICP$ to the target $PINC$. Both goals must be minimized. a_1 weights the relative importance of the first goal versus the second. The meaning of term $\max\{PINC - PICP, 0\}$ in Eq. 8 is that we are interested in intervals that get as close to the target $PINC$ as possible, but it is not important to go beyond that value. MIRAR-RAM:For instance, if the target is $PINC = 0.95$, a PI with $PICP = 0.94$ would be considered better than another with $PICP = 0.97$. Otherwise, the optimization process might return PIs with higher-than-necessary $PICP$, but at the cost of very large AIW.

For the multi-objective formulation of the problem, the two goals to be optimized are $PICP$ and AIW. In order to have two goals to be minimized, the problem is formulated as in Eq. 9, where the second goal is $\alpha = 1 - PICP$.

$$F_{MOPSO} = \{AIW, 1 - PICP\} = \{AIW, \alpha\} \tag{9}$$

It can be noticed that the target $PINC$ is not involved in the definition of the objective space (Eq. 9), because selecting the solution closest to the target is done **after** the optimization procedure has been run and generated the Pareto front. Figure 2 displays an actual Pareto front obtained from one of our experiments. This front represents the optimal tradeoffs between coverage probability ($PICP$) and width (AIW) and the user can select the solution with the desired target $PINC$ value. If the target $PINC$ is 0.9, the point in the Pareto front closest to the 0.9-horizontal line would be selected. In this case, this would be the solution with $PICP = 0.9020$ and $AIW = 0.3186$. This solution corresponds to a MOPSO particle that codifies the weights of the MLP. It is important to remark that our method uses the training dataset in order to carry out the selection process described (i.e. $(PICP, AIW)$ pairs in Figure 2 have been computed with the training dataset). The selected MLP solution might have different $(PICP, AIW)$ values in the test dataset, but if the MLP models do not overfit, they should be similar. In this particular case, the $PICP$ and AIW of the selected MLP on the test set are 0.9086 and 0.3276, respectively, very close to the training values.

5 Data

The data available from the Kaggle website has been provided by the American Meteorological Society [1] in the context of a solar energy prediction contest (McGovern et al, 2015). The goal is to predict the total daily incoming solar energy, measured in $J \times m^{-2}$, at 98 sites of the Oklahoma Mesonet network, which covers a surface of, approximately, 180000 square kilometers. The input data for each day corresponds to the output of the numerical weather prediction model GEFS (global ensemble forecast system) using 11 ensemble members and 5 forecast timesteps from 12 to 24 hours in 3 hour increments. Each ensemble member produces outputs for 15 different meteorological variables for each timestep and each point of a 16×9 uniform land-surface grid, with a spatial resolution of about 90 km, including the State of Oklahoma and surrounding areas. Figure 3 shows the GEFS grid nodes (blue points) and the 98 Oklahoma Mesonet sites (red points). Some of the meteorological variables used are the following: accumulated precipitation ($kg.m^{-2}$), air pressure (Pa), downward and upward shortwave/longwave radiation ($W.m^{-2}$), cloud cover (%), temperature (K), etc. A more detailed information can be found in [1].
Thus, the number of attributes for each grid node is $11 \times 5 \times 15 = 875$. In order

[1] https://www.kaggle.com/c/ams-2014-solar-energy-prediction-contest

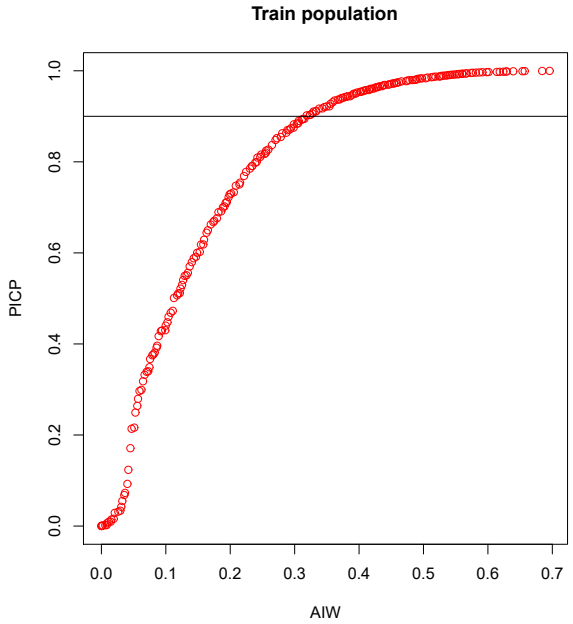

Fig. 2 Population of solutions obtained by MOPSO with the training set

to reduce the amount of data, we have combined the 11 ensemble members of the GEFS system using the mean of each meteorological variable at each timestep, as we justified in our previous work (Martin et al, 2015). Then, the number of attributes for each node is reduced to $5 \times 15 = 75$. In that work we also showed that four grid nodes where enough to obtain reasonable results in the prediction of the daily solar radiation in the nodes of the Oklahoma mesonet network. For this reason, in this work we have used the four grid nodes around the mesonet site. Then, the number of input variables used is $5 \times 15 \times 4 = 300$.

Data has been collected everyday from 1994 to 2007 (5113 days) in association with the corresponding accumulated incoming solar energy, which is the attribute to be predicted. This accumulated incoming solar energy (in $J \times m^{-2}$) has been calculated by summing the solar energy measured by a pyranometer at each mesonet site every 5 minutes, from the sunrise to 23:55 UTC of the corresponding date. From the total input-output available data covering 14 years for each station, we have used the period 1994-2005 as the training set (4383 days), reserving the period 2006-2007 (730 days) for the testing set.

In order to validate the performance of the proposal to estimate PI, the experiments have been run for two sites from the 98 sites available, situated at different latitude and longitude locations, concretely station 1 (at latitude

34.808330 and longitude -98.023250) and station 98 (at latitude 36.518060 and longitude -96.342220), although any Mesonet station could be used.

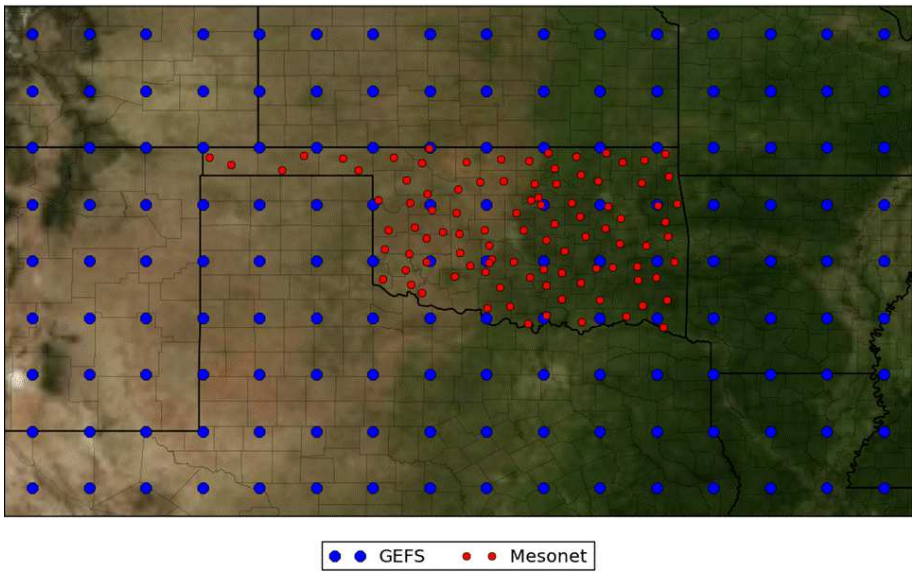

Fig. 3 GEFS grid points and mesonet stations (McGovern et al, 2015)

6 Experimentation

In this section, the experimental results about the estimation of PIs using the single and multi-objective approaches are shown. In order to compare with a baseline, the ensemble method Gradient Boosted Regression (GBR) (Friedman, 2000, 2002; Schonlau , 2005) has also been used for estimating PIs. GBR is a recent machine learning technique that has shown considerable success in regression tasks. It produces a prediction model in the form of an ensemble of weak prediction models, typically regression trees. GBR uses boosting to construct the ensemble by sequentially adding models, so that each model focuses on the errors of the previous one. GBR can also be used for quantile regression (or quantile estimation) (Kriegler and Berk, 2007; Zheng , 2012). We have followed the approach recommended in (Meinshausen, 2006): in order to compute a PI with a probability coverage $PINC$, GBR is used for estimating two quantiles, Lower=$\frac{PINC}{2}$ and Upper=$1 - \frac{PINC}{2}$. Thus, the PI with a coverage of $PINC$ is $[Lower, Upper)$.

Table 1 Parameter values explored for each method

Method	Parameter	Values
MOPSO	Hidden Neurons	$\{15, 30, 60, 80, 100, 120\}$
	Iterations	$\{1000, 2000, 3000, 4000\}$
PSO	Hidden Neurons	$\{15, 30, 60, 80, 100, 120\}$
	Iterations	$\{500, 1000, 1500, 2000, 2500, 3000, 3500, 4000\}$
	a_1	$\{0.2, 0.3, 0.4\}$
GBR	Trees	$\{100, 500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000\}$
	Depth	$\{2, 4, 6, 8, 19, 12, 14\}$
	Shrinkage	$\{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$

Before obtaining the results for the estimation of PIs with the different methods (multi-objective, single-objective and quantile estimation by GBR), we have performed an extensive task of experimentation in order to decide the main parameters used for each of these methods, and to study the sensitivity of the results to the values of these parameters. This study is carried out using data from station 1. The best configurations of parameters will be used to obtain the results for the second station (station 98).

This section is organized as follows: in subsection 6.1 we explain the procedure used to select the optimal combinations of parameters for each of the methods; in 6.2 we report and analyze the results of those best combinations of parameters for all methods, on two different measuring stations and the test datasets; in subsection 6.3 we study the sensitivity of the results to the main parameters. The last subsection reports the execution times for each of the methods. In the text of this section, MOPSO refers to the multi-objective method, PSO to the single-objective one and, GBR to quantile estimation by means of Gradient Boosting.

6.1 Parameter selection

We have performed a series of experiments with different combinations of parameters, in order to select the most suitable configuration for each method. Table 1 displays the parameter values considered for each method. For MOPSO and PSO, the main parameters are the number of hidden neurons and the number of iterations carried out by the evolutionary algorithm. Also for PSO, a_1 that weights the two goals to optimize. For GBR, its three main parameters are the number of trees in the ensemble, the maximum depth of each tree, and the shrinkage (or learning rate).

We use a common procedure for parameter selection called grid search, where all possible combinations of parameters are evaluated. They are evaluated by generating a model using the training data, and evaluating it on the validation set. The combination that attains the best validation value is the one finally selected, as it is expected to generalize well over the test set. In order to get more accurate results, each combination of parameters has been run 5 times, with different random seeds, and results are averaged. The previous procedure has been performed for each of three different $PINC$ values used

in this article: 0.99, 0.95 and 0.90, except for MOPSO, where $PINC$ is not a parameter but a decision criterion used to select a solution from the Pareto front. For both PSO and MOPSO typical values found in literature are used, for parameters such as the inertia coefficient ($w = 0.4$), and the mutation probability($pMut = 0.5$). The population size was fixed to 100 particles and the archive size of MOPSO was 500.

In order to select the best set of parameters for each of the methods, we have used the following measures:

- **Multi-objective approach**: the best combination of MOPSO parameters is selected using the hyper-volume metric (Zitzler and Thiele , 1999), averaged for each of the 5 runs of each experiment. Hyper-volume is typically used in multi-objective approaches to evaluate the quality of the final Pareto fronts. In the present 2-objective case, the hyper-volume is the area covered by the Pareto front; the higher, the better. Two different values of hyper-volume have been computed: HV is the total hyper-volume, and HV_2 is the hyper-volume in the section of the Pareto front related to the high-coverage PIs ($PICP \geq 0.90$), that will be used later.
- **Single-objective approach and quantile estimation by GBR**: the average value of $ACE = |PINC - PICP|$ has been used. That is, a combination of parameters is preferred if it results in a solution with $PICP$ closest to the nominal $PINC$.

Following the common practice for parameter selection in machine learning, all these measures were calculated on a validation set that is independent from both the training and the test data sets. As we have mentioned in Section 5, the period from 1995 to 2005 has been used as training. As validation set, data from 2004 and 2005 has been extracted from the training set. This choice allows us to select configurations that are independent of the test data, but are expected to generalize well to provide good results on the test set. Note that test data is composed of measures from the two most recent years (2006 and 2007).

In Table 2, we provide the five best combinations of parameters for MOPSO in terms of HV_2 and HV on the validation set. We have selected as best result the experiment with 80 neurons and 4000 iterations. As it can be observed, it ranked first on the restricted hyper-volume (HV_2) and second in total hyper-volume (HV). It can also be observed that the values of HV_2 and HV are very similar in the five best experiments.

With respect to the number of iterations for MOPSO, Table 2 shows that 4000 is the best value. Given that this is the largest value that has been evaluated, it could be considered that perhaps better results could be obtained by using more iterations. Figure 4 displays the evolution of the validation-set Pareto front, as the number of iterations increases (and for MLPs with 80 neurons). Fronts are shown for 1000, 2000 and 4000 iterations. Results show that fronts for 2000 and 4000 iterations are fairly similar, while for 1000 iterations convergence of the front is not yet complete. Therefore we conclude that

Table 2 Best five combinations of parameters for MOPSO on the the validation set

Parameters		Metric	Parameters		Metric
Neurons	Iterations	HV_2 ($PICP \geq 0.9$)	Neurons	Iterations	HV
80	4000	0.0553088226	120	4000	0.8350620363
80	3000	0.0551441698	80	4000	0.834025399
120	4000	0.0547833572	60	4000	0.8337651769
100	4000	0.0546995848	120	3000	0.8336586572
80	2000	0.0546599585	80	3000	0.8332798968

the evolution has almost converged and that further increasing the number of iterations is not expected to improve the results.

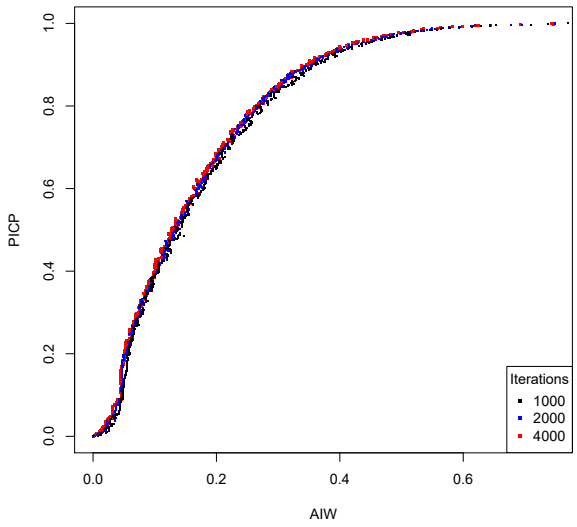

Fig. 4 Evolution of the validation-set Pareto fronts for MOPSO.

In Table 3 we show the five combinations that gave the best results (in the validation set) for PSO approach and GBR quantile estimation. Let's remember that for both approaches, different experiments must be carried out for each desired $PINC$ value (in this work 0.90, 0.95 and 0.99). The lower the ACE metric, the better the method approximates those nominal values. For PSO, for each value of $PINC$, the best result is achieved by different combinations of the parameters. However, some similarities can be observed. For PSO, the best results are always for $a_1 = 0.2$. This was expected as this value gives more importance to the $PINC$ objective, and thus works towards smaller values of ACE. Also for PSO, none of the best experiments required more than 500 iterations, despite allowing the system to run until 4000 itera-

Table 3 Best five combinations of parameters for PSO and GBR quantile estimation on the the validation set

	PSO				GBR			
	Parameters			Metric	Parameters			Metric
$PINC$	Neurons	Iters.	a_1	ACE	Trees	Depth	Shrink.	ACE
0.90	60	500	0.2	6.32E-003	5000	12	0.001	5.75E-004
	120	500	0.2	8.07E-003	500	14	0.01	1.40E-003
	100	500	0.2	8.89E-003	1000	12	0.005	1.40E-003
	80	500	0.2	1.00E-002	4500	14	0.001	1.50E-003
	30	500	0.2	1.09E-002	5000	14	0.001	1.56E-003
0.95	100	500	0.2	2.83E-003	5000	10	0.001	6.70E-004
	80	500	0.2	3.63E-003	1500	4	0.005	6.98E-004
	30	500	0.2	4.50E-003	5000	12	0.001	9.17E-004
	60	500	0.2	6.14E-003	2500	2	0.005	9.44E-004
	100	1000	0.2	7.48E-003	2000	2	0.005	9.99E-004
0.99	15	500	0.2	6.69E-003	4000	10	0.001	4.24E-004
	80	500	0.2	8.60E-003	3500	10	0.001	6.98E-004
	15	1000	0.2	1.11E-002	1000	8	0.005	7.36E-004
	120	500	0.2	1.16E-002	4500	4	0.005	8.02E-004
	80	1000	0.2	1.22E-002	5000	8	0.001	8.40E-004

tions. For GBR, values of the parameters were also different among the three series of experiments. However, it was clear that $Shrinkage = 0.001$ always produced the best results. Also, the best experiments used a high number of trees (either 4000 or 5000). Finally, the depth parameter in these experiments was either 10 or 12. In both PSO and GBR, variations in ACE are in fact not large, so different combinations of parameters should provide similar results.

MIRAR-RAM: Similar to MOPSO, for PSO we also analyze the evolution of the fitness (Eq. 8) as the number of iteration increases. Figures 5, 6 and 7 show the values of fitness for both the training and the validation sets at 500, 1000, 1500, 2000, 2500, 3000, 3500 and, 4000 iterations. It is observed how fitness is quite stable for the train and validation after 500 iterations. For $PINC = 0.9$ and $PINC = 0.95$ a certain degree of over-fitting is observed after 500 iterations. Therefore, for PSO we can also concluded that the evolution has converged and the best results are archived at 500 iterations.

6.2 Experimental results

In this section we report and compare the results of the three methods, on the testing set, for the two mesonet sites mentioned in Section 5, using the best configuration of parameters for each method determined in the previous subsection.

For all the experiments, the goal is to achieve a target $PINC$ value (in this work 0.9, 0.95, and 0.99). Single-objective PSO and quantile estimation based in GBR must be run for each desired target value, while MOPSO is run only once and then the solution for the desired value is selected from the Pareto front.

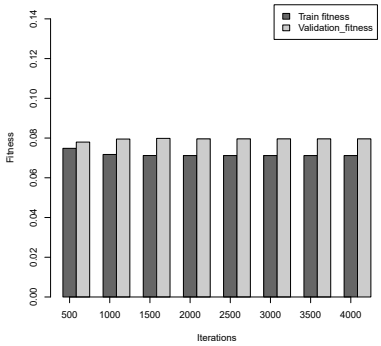

Fig. 5 Evolution of fitness for PINC value 0.90

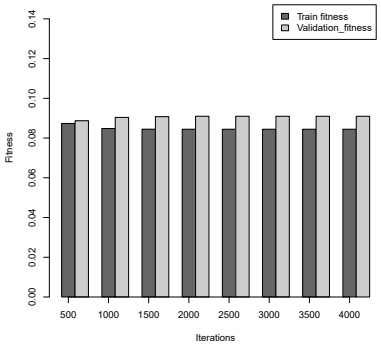

Fig. 6 Evolution of fitness for PINC value 0.95

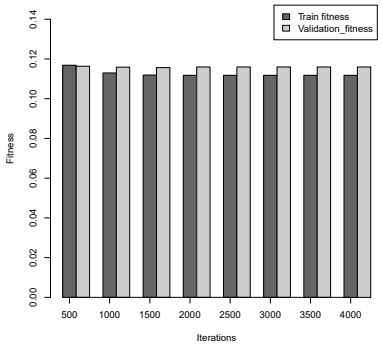

Fig. 7 Evolution of fitness for PINC value 0.99

Table 4 shows the $PICP$ and AIW results, grouped by $PINC$, for both the training and testing sets for Stations 1 and 98. For station 1 it can be seen that, when a $PINC$ level of 0.90 is required, MOPSO obtains for the testing set a $PICP$ value of 0.913 with an AIW of 0.331. PSO and GBR also fulfill the $PINC$ requirement although the obtained AIWs are slightly higher (0.383 and 0.362, respectively).

For the $PINC = 0.95$ level, we observe that both MOPSO and PSO reach the required 0.95 value, while GBR is close ($PICP = 0.944$). But GBR interval width is broader than either MOPSO or PSO (0.516 vs. 0.412 and 0.440).

For the hardest $PINC$ level, 0.99, we can observe that GBR reaches the target for the testing set, but at the expense of AIW, which is very broad ($AIW = 0.784$). MOPSO is very close to the nominal coverage (obtaining $PICP = 0.984$) and also obtains a narrower PI compared to GBR ($AIW = 0.552$). Finally, PSO also obtains reasonable values for $PINC$ and AIW (0.981 and 0.567, respectively), but worse than MOPSO.

Table 4 Comparison by $PINC$ for Stations 1 and 98 with different methods

		Station 1				Station 98			
		Train		Test		Train		Test	
$PINC$	Method	AIW	$PICP$	AIW	$PICP$	AIW	$PICP$	AIW	$PICP$
0.9	MOPSO	0.318	0.900	0.331	0.913	0.353	0.899	0.351	0.911
	PSO	0.374	0.900	0.383	0.908	0.446	0.900	0.450	0.917
	GBR	0.364	0.947	0.362	0.904	0.382	0.949	0.378	0.893
0.95	MOPSO	0.398	0.950	0.412	0.950	0.433	0.951	0.445	0.960
	PSO	0.436	0.950	0.448	0.950	0.485	0.950	0.490	0.957
	GBR	0.519	0.964	0.516	0.944	0.531	0.968	0.527	0.939
0.99	MOPSO	0.539	0.990	0.552	0.984	0.590	0.990	0.602	0.996
	PSO	0.557	0.983	0.567	0.981	0.680	0.983	0.681	0.987
	GBR	0.782	0.983	0.784	0.990	0.805	0.985	0.806	0.984

In the right side of Table 4, results for Station 98 are shown. For $PINC$ values 0.90 and 0.95, both MOPSO and PSO fulfill the nominal level, although the AIW obtained by MOPSO (0.351 and 0.445) are narrower than the ones obtained by PSO (0.450 and 0.490). For $PINC$ values 0.90 and 0.95, GBR does not reach the nominal level in testing (0.893 and 0.939, respectively) and the AIW are in both cases broader than those of MOPSO. Finally, for the most strict $PINC$ level (0.99), the only method which reaches the nominal value is MOPSO (0.996) compared to PSO (0.987) and GBR (0.984). Besides, MOPSO obtains the best AIW (0.602) compared to PSO (0.681) and GBR (0.806).

Summarizing results for both stations, MOPSO usually obtains $PICP$ values very close to the nominal required value, in addition to providing the narrowest PIs.

Figures 8, 9 and 10 display the test results for all methods on the objective space, for each of the $PINC$ values and for each of the five runs. The background plot shows the cloud of points representing the five test-set Pareto fronts. The horizontal line displays the target $PINC$s. The solution from the MOPSO experiment that is closer to this $PINC$ is selected as the solution for that experiment. Note that in this case, instead of showing the average results of the five experiments, as in Table 4, we show each of the five experiments independently.

These figures show that, while for $PINC = 0.90$ GBR gives a solution that is comparable to MOPSO, and better than PSO, as we increase $PINC$ GBR generates very wide intervals (higher AIW) that are very far from the solutions of MOPSO. It is also clear that solutions provided by PSO are close to solutions by MOPSO only for the smallest value of $a_1 = 0.2$, but those solutions have also wider AIW than the corresponding solutions of MOPSO.

Finally and for illustrative purposes, we can see in Figures 11, 12 and 13 the predictions of the upper and lower limits of the intervals and the actual output value (radiation) for each day of one year within the two year period of the testing set. These predictions on the testing set correspond to one of the five MOPSO solutions selected for $PINC = 0.9$, $PINC = 0.95$ and, $PINC = 0.99$, respectively. As we explained before, each solution is taken from

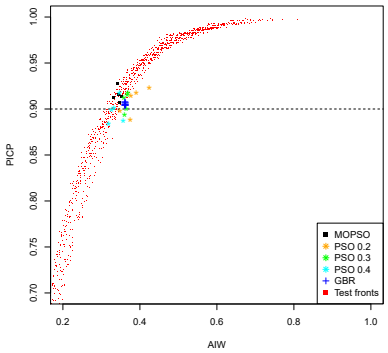

Fig. 8 Comparison of methods for $PINCP = 0.90$

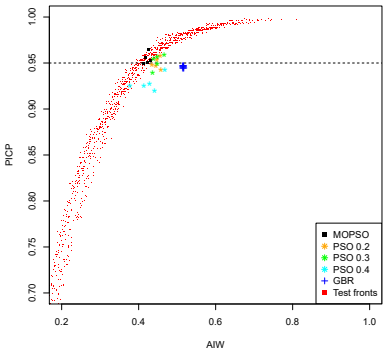

Fig. 9 Comparison of methods for $PINCP = 0.95$

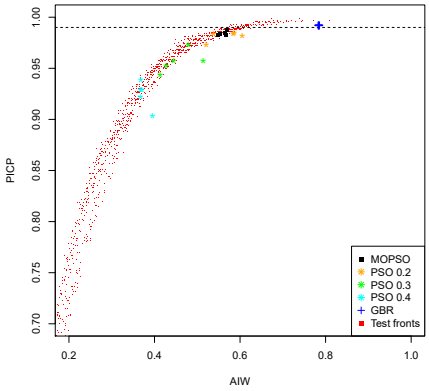

Fig. 10 Comparison of methods for $PINCP = 0.99$

the corresponding Pareto front that was built by MOPSO using 80 neurons and 4000 iterations. We can see that the actual radiation values for each day fit well inside the intervals formed by the predictions of the upper and lower limits of the intervals for each day. In order to make the figure more clear, we have chosen one year inside the two-year period of the testing set.

6.3 Sensitivity to parameters

In subsection 6.1, the best combination of parameters was exhaustively determined for each of the approaches using the validation set. In subsection 6.2, these best configurations were evaluated on the test set. Those can be considered as our main results. In the present subsection, we show whether results

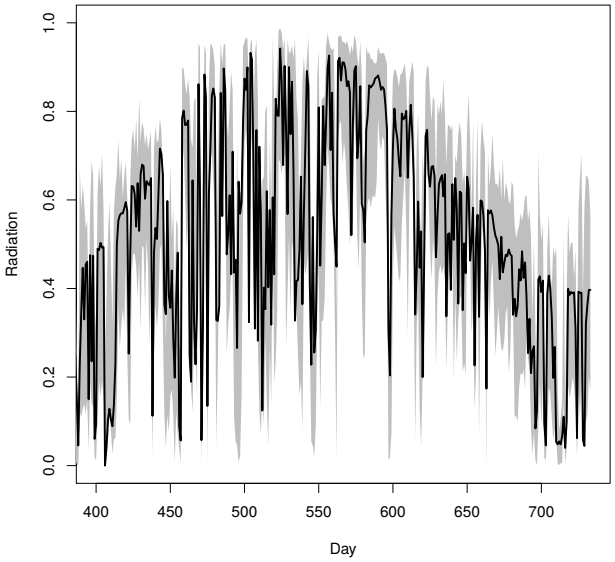

Fig. 11 Prediction of the upper and lower limits of the interval and the actual output value, corresponding to one year period within the testing set for a $PINC = 0.90$. MOPSO solution with 80 neurons and 4000 iterations.

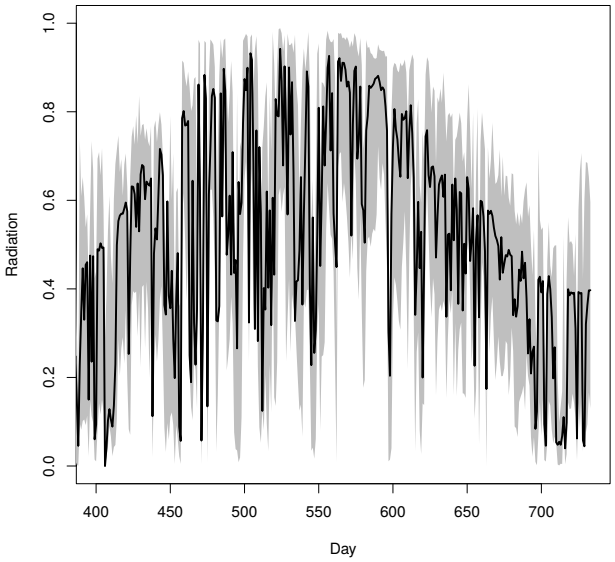

Fig. 12 Prediction of the upper and lower limits of the interval and the actual output value, corresponding to one year period within the testing set for a $PINC = 0.95$. MOPSO solution with 80 neurons and 4000 iterations.

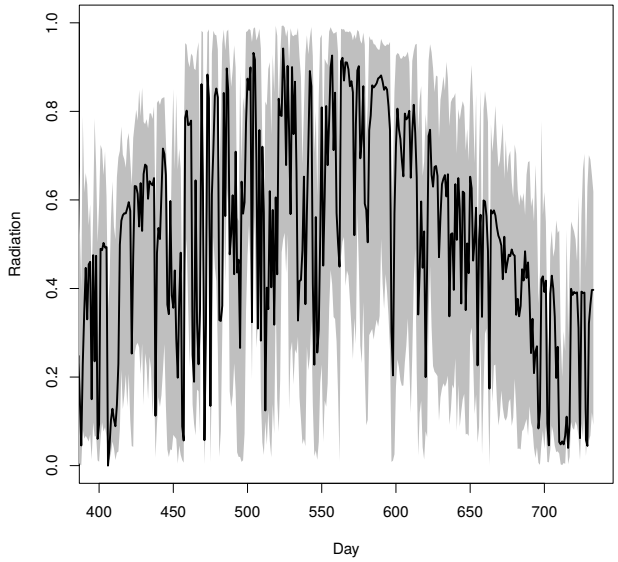

Fig. 13 Prediction of the upper and lower limits of the interval and the actual output value, corresponding to one year period within the testing set for a $PINC = 0.99$. MOPSO solution with 80 neurons and 4000 iterations.

depend strongly on the precise parameter values selected for MOPSO and PSO. This is done for two of the most important parameters: the number of hidden neurons (for both MOPSO and PSO) and the a_1 coefficient for PSO (see Eq. 8), that controls the tradeoff between width and coverage.

Figures 14 and 15 show how AIW and $PICP$ for MOPSO, change as the number of hidden neurons in the MLP increases. AIW and $PICP$ shown in those figures are computed on the test data set. Also, each value displayed in the figures uses the optimal value found out for the rest of the parameters (for instance, for 80 hidden neurons, the value displayed uses 4000 iterations, because that was the optimal value found out when using 80 neurons (see Table 2)). Each figure contains three lines corresponding to the different values of $PINC$ (0.90, 0.95 and 0.99). It is observed that both $PICP$ and AIW do not change much once the number of neurons reaches 80 (and beyond). For smaller number of neurons, while $PICP$ is near the nominal value, this is achieved at the cost of an increase in the value of AIW.

Figures 16 and 17 show the same information than figures 14 and 15, but regarding the PSO single-objective approach. For $PICP = 0.99$ the number of neurons affects the AIW value to a greater extent. In this case, results show an increase in AIW and results suggests that the number of neurons should be set at least to 60 and no more than 100. For $PINC = 0.90$ and $PINC = 0.95$ the effect in the value of AIW is minor and results show that the number of hidden neurons should also be set at least to 60. The values of $PICP$ do not show great variation respect to the number of hidden neurons.

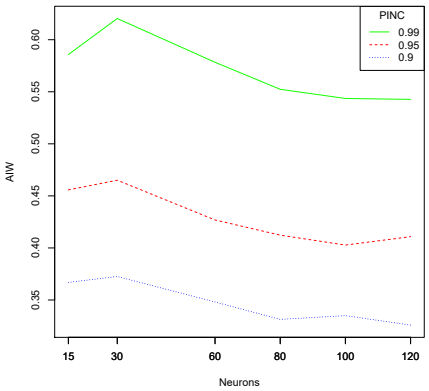

Fig. 14 MOPSO: Sensitivity of AIW to the number of neurons

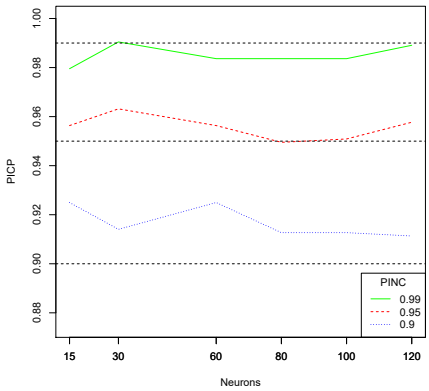

Fig. 15 MOPSO: Sensitivity of $PICP$ to the number of neurons

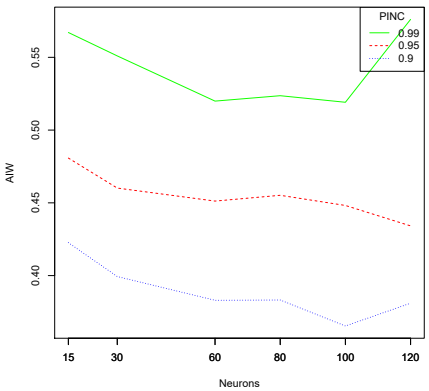

Fig. 16 PSO: Sensitivity of AIW to the number of neurons

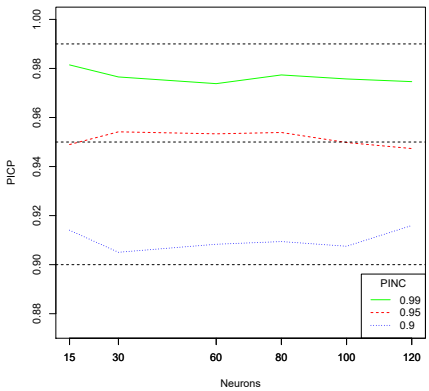

Fig. 17 PSO: Sensitivity of $PICP$ to the number of neurons

Finally, the sensitivity to the coefficient a_1 in PSO single-objective (see Eq. 8) is shown in Figures 14 and 19. As we have already mentioned, this term balances the relative importance of both AIW and $PICP$ as optimization objectives. Results show that AIW is better and $PICP$ worse for higher values of a_1. This effect is much more important for the experiments with $PICP = 0.99$.

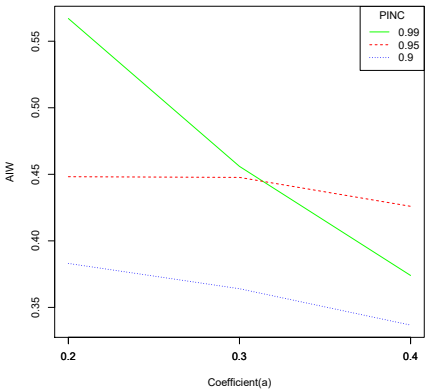

Fig. 18 PSO: Sensitivity of AIW to the fitness coefficient (a_1)

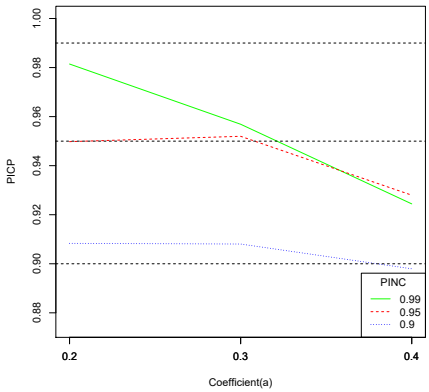

Fig. 19 PSO: Sensitivity of $PICP$ to the fitness coefficient (a_1)

Table 5 Average execution times, in minutes, for MOPSO (4000 iterations) and PSO (500 iterations)

Neurons	MOPSO (mins.)	PSO (mins.)
15	99.796	13.046
30	163.716	19.910
60	263.665	73.312
80	333.883	75.913
100	389.715	50.021
120	431.501	127.682

6.4 Execution times

To give an idea of the order of magnitude of the computational costs of the different approaches, we provide execution times in Table 5 and 6. All experiments were performed on an Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz, and times reported correspond to a single core.

Differences between MOPSO and PSO are mainly due to the number of iterations. We report values for 4000 iterations for MOPSO and 500 iterations for PSO, because those are the most appropriate values for each approach. This explains the higher computational cost observed for MOPSO. In both cases, time increases with the size of the neural network. The computational cost of quantile estimation based on GBR is generally less than the computational time required by MOPSO and PSO. However, it must be noticed that PSO and GBR must be run for every value of $PINC$, while using MOPSO one single run is enough.

Table 6 Average execution times, in minutes, for GBR

Trees	Depth	Shrinkage	Time (mins.)
4000	10	0.001	20.110
4000	10	0.001	25.138
5000	12	0.001	30.375

7 Conclusions

In this paper a multi-objective evolutionary approach is proposed for the construction of PIs using a MLP. The inputs to the network are meteorological variable forecasts for time t and the outputs are the PI upper and lower bounds of the solar energy generated at time t. The MLP is trained using MOPSO to minimize the width of PIs and to maximize their coverage probability (or reliability). More concretely, the proposal computes models to estimate upper and lower bounds of PIs for every sample input such that the average interval width (AIW) is minimized and the prediction interval coverage probability ($PICP$) is greater or equal than a nominal level, the prediction interval nominal confidence ($PINC$). Both objectives are conflicting, in the sense that improving one of them usually degrades the other one. Our proposal takes advantage of multi-objective evolutionary algorithms, providing a set of solutions (the non-dominated solution set or Pareto front) that represents all the optimal tradeoffs between the two objectives, from where the user can choose according to his needs (typically, the solution that achieves the target $PINC$). We show that the multi-objective approach is advantageous over the single-objective formulation because the former does not require to weight the two objectives in advance. Rather, both of them are optimized simultaneously, providing a set of solutions (the Pareto front) in a single run, and delaying the selection of the most appropriate one after the optimization process has taken place. Additionally, the user can observe the trade-offs between the two objectives by analyzing the Pareto front, and explore the consequences on one of the objectives of changes made on the other one.

The proposal has been validated using data related to solar energy generation, where meteorological variable forecasts were obtained from the Global Forecast System, and the accumulated radiation was measured at two of the Oklahoma mesonet sites. Results have been compared with a single-objective PSO approach and with non-linear quantile regression by means of the GBR method. Different $PINC$ target values have been tested. When intervals are estimated using single-PSO or GBR, a run for every value of $PINC$ has been carried out, while MOPSO was run only once, as explained in the previous paragraph. In this case, the set of solutions (the Pareto front) can be used for any possible target $PINC$.

A complete set of experiments have been carried out in order to select the best parameter configuration for all the methods. The final experimental results show that the multi-objective approach provides similar or better results than single-PSO and GBR, depending on the desired value of $PINC$. For the

most difficult target value, $PINC = 0.99$, MOPSO obtains the probability coverage very close to the target $PINC$ while at the same time, providing the narrowest intervals. When methods provide similar $PICP$ values, MOPSO provides the best value of AIW.

Funding: This work has been funded by the Spanish Ministry of Science under contract ENE2014-56126-C2-2-R (AOPRIN-SOL project).

References

Afshari-Igder M, Niknam T, Khooban MH (2016) Probabilistic wind power forecasting using a novel hybrid intelligent method. Neural Computing and Applications, 1–13.

Alessandrini S, Delle Monache L, Sperati S, Cervone G (2015). An analog ensemble for short-term probabilistic solar power forecast. Applied energy, 157, 95–110.

Alonso A, Torres A, Dorronsoro JR (2015) Random forests and gradient boosting for wind energy prediction. In: Hybrid Artificial Intelligent Systems - 10th International Conference, HAIS 2015, Bilbao, Spain, June 22-24, 2015, Proceedings, pp 26–37, DOI 10.1007/978-3-319-19644-2_3

Ameli K, Alfi A, Aghaebrahimi M (2016). A fuzzy discrete harmony search algorithm applied to annual cost reduction in radial distribution systems. Engineering Optimization, 48(9), 1529–1549.

Arab A, Alfi A (2015). An adaptive gradient descent-based local search in memetic algorithm applied to optimal controller design. Information Sciences, 299, 117–142.

Arqub OA, Abo-Hammour Z (2014). Numerical solution of systems of second-order boundary value problems using continuous genetic algorithm. Information sciences, 279, 396–415.

Babalk A (2017). A novel metaheuristic for multi-objective optimization problems. Information Sciences: an International Journal, 402(C), 124–148.

Bacher P, Madsen H, Nielsen HA (2009) Online short-term solar power forecasting. Solar Energy 83(10):1772–1783

Bracale A, Caramia P, Carpinelli G, Di Fazio AR, Ferruzzi G (2013) A bayesian method for short-term probabilistic forecasting of photovoltaic generation in smart grid operation and control. Energies 6(2):733–747

Bremnes JB (2004) Probabilistic wind power forecasts using local quantile regression. Wind Energy 7(1):47–54

Chai S, Xu Z, Wong WK (2015) Optimal granule-based PIs construction for solar irradiance forecast. IEEE Transactions on Power Systems

Chu Y, Li M, Pedro HT, Coimbra CF (2015) Real-time prediction intervals for intra-hour DNI forecasts. Renewable Energy 83:234–244

Coello CA, Toscano G, Salazar M (2004) Handling multiple objectives with particle swarm optimization. IEEE Transactions on Evolutionary Computation 8(3):256–279

Costa A, Crespo A, Navarro J, Lizcano G, Madsen H, Feitosa E (2008) A review on the young history of the wind power short-term prediction. Renewable and Sustainable Energy Reviews 12(6):1725–1744

Deb K (2001) Multi-objective optimization using evolutionary algorithms, vol 16. John Wiley & Sons

Friedman JH (2000) Greedy function approximation: A gradient boosting machine. Annals of Statistics 29:1189–1232

Friedman JH (2002) Stochastic gradient boosting. Computational Statistics & Data Analysis 38(4):367–378

Gala Y, Pascual AF, García JD, Dorronsoro JR (2016) Hybrid machine learning forecasting of solar radiation values. Neurocomputing 176:48–59, DOI 10.1016/j.neucom.2015.02.078

Juban J, Fugon L, Kariniotakis G (2007). Probabilistic short-term wind power forecasting based on kernel density estimators. In European Wind Energy Conference and exhibition, EWEC 2007.

Kavousi-Fard A, Khosravi A, Nahavadi S (2014) A novel fuzzy multi-objective framework to construct optimal prediction intervals for wind power forecast. In: Neural Networks (IJCNN), 2014 International Joint Conference on, IEEE, pp 1015–1019

Kennedy J, Eberhart R, YShi (2001) Swarm intelligence. Morgan Kaufmann Publishers, San Francisco

Khosravi A, Nahavandi S, Creighton D, Atiya AF (2011). Lower upper bound estimation method for construction of neural network-based prediction intervals. IEEE Transactions on Neural Networks, 22(3), 337–346.

Khosravi A, Nahavandi S (2013) Combined nonparametric prediction intervals for wind power generation. Sustainable Energy, IEEE Transactions on 4(4):849–856

Knowles JD, Corne DW (2000) Approximating the nondominated front using the pareto archived evolution strategy. Evolutionary Computation 8(2):149–172

Kriegler B, Berk R (2007). Boosting the quantile distribution: A cost-sensitive statistical learning procedure. Department of Statistics, UCLA, working paper.

Linares-Rodriguez A, Quesada-Ruiz S, Pozo-Vazquez D, Tovar-Pescador J (2015) An evolutionary artificial neural network ensemble model for estimating hourly direct normal irradiances from meteosat imagery. Energy 91:264–273

Marin LG, Valencia F, and Saez D (2016, July). Prediction interval based on type-2 fuzzy systems for wind power generation and loads in microgrid control design. In Fuzzy Systems (FUZZ-IEEE), 2016 IEEE International Conference, 328–335

Marquez R, Coimbra CF (2011) Forecasting of global and direct solar irradiance using stochastic learning methods, ground experiments and the NWS

database. Solar Energy 85(5):746–756
Marquez R, Pedro HT, Coimbra CF (2013) Hybrid solar forecasting method uses satellite imaging and ground telemetry as inputs to ANNs. Solar Energy 92:176–188
Martin R, Aler R, Valls JM, Galvan IM (2015) Machine learning techniques for daily solar energy prediction and interpolation using numerical weather models. Concurrency and Computation: Practice and Experience pp n/a–n/a, DOI 10.1002/cpe.3631, cpe.3631
McGovern A, Gagne DJ, Basara J, Hamill TM, Margolin D (2015) Solar energy prediction: an international contest to initiate interdisciplinary research on compelling meteorological problems. Bulletin of the American Meteorological Society 96(8):1388–1395
Meinshausen N (2006) Quantile regression forests. The Journal of Machine Learning Research 7:983–999
Mellit A (2008) Artificial intelligence technique for modelling and forecasting of solar radiation data: a review. International Journal of Artificial intelligence and soft computing 1(1):52–76
Mellit A, Pavan AM (2010) A 24-h forecast of solar irradiance using artificial neural network: Application for performance prediction of a grid-connected PV plant at trieste, italy. Solar Energy 84(5):807–821
Mousavi Y, Alfi A (2015). A memetic algorithm applied to trajectory control by tuning of Fractional Order Proportional-Integral-Derivative controllers. Applied Soft Computing, 36, 599–617.
Nielsen HA, Madsen H, Nielsen TS (2006) Using quantile regression to extend an existing wind power forecasting system with probabilistic forecasts. Wind Energy 9(1-2):95–108
Okumus I, Dinler A (2016) Current status of wind energy forecasting and a hybrid method for hourly predictions. Energy Conversion and Management, 123, 362–371.
Pahnehkolaei SMA, Alfi A, Sadollah A, Kim JH (2017). Gradient-based Water Cycle Algorithm with evaporation rate applied to chaos suppression. Applied Soft Computing, 53, 420–440.
Pedro HT, Coimbra CF (2012) Assessment of forecasting techniques for solar power production with no exogenous inputs. Solar Energy 86(7):2017–2028
Pinson P (2013) Wind energy: Forecasting challenges for its operational management. Statistical Science 28(4):564–585
Pinson P, Kariniotakis G (2010) Conditional prediction intervals of wind power generation. Power Systems, IEEE Transactions on 25(4):1845–1856
Raquel CR, Naval PC (2005) An Effective use of Crowding Distance in Multiobjective Particle Swarm Optimization. Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2005), Washington, D.C., June 25-29:257–264
Schonlau M (2005) Boosted regression (boosting): An introductory tutorial and a stata plugin. Stata Journal 5(3):330
Sideratos G, Hatziargyriou ND (2012) Probabilistic wind power forecasting using radial basis function neural networks. Power Systems, IEEE Transac-

tions on 27(4):1788–1796
Talbi EG (2009) Metaheuristics: from design to implementation, vol 74. John Wiley & Sons
Wan C, Xu Z, Pinson P (2013). Direct interval forecasting of wind power. IEEE Transactions on Power Systems, 28(4), 4877-4878.
Wan C, Xu Z, Pinson P, Dong ZY, Wong KP (2014) Probabilistic forecasting of wind power generation using extreme learning machine. Power Systems, IEEE Transactions on 29(3):1033–1044
Wan C, Lin J, Song Y, Xu Z, Yang G (2017). Probabilistic forecasting of photovoltaic generation: an efficient statistical approach. IEEE Transactions on Power Systems, 32(3), 2471–2472.
Zhang Y, Wang J, Wang X (2014) Review on probabilistic forecasting of wind power generation. Renewable and Sustainable Energy Reviews 32:255–270
Zheng S (2012). QBoost: Predicting quantiles with boosting for regression and binary classification. Expert Systems with Applications, 39(2), 1687–1697.
Zitzler E, Thiele L (1999). Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. Trans. Evol. Comp 3 (4): 257271. http://dx.doi.org/10.1109/4235.797969.