Neural network river forecasting with multi-objective fully informed particle swarm optimization

Riccardo Taormina and Kwok-wing Chau

ABSTRACT

In this work, we suggest that the poorer results obtained with particle swarm optimization (PSO) in some previous studies should be attributed to the cross-validation scheme commonly employed to improve generalization of PSO-trained neural network river forecasting (NNRF) models. Cross-validation entails splitting the training dataset into two, and accepting particle position updates only if fitness improvements are concurrently measured on both subsets. The NNRF calibration process thus becomes a multi-objective (MO) optimization problem which is still addressed as a single-objective one. In our opinion, PSO cross-validated training should be carried out under an MO optimization framework instead. Therefore, in this work, we introduce a novel MO variant of the swarm optimization algorithm to train NNRF models for the prediction of future streamflow discharges in the Shenandoah River watershed, Virginia (USA). The case study comprises over 9,000 observations of both streamflow and rainfall observations, spanning a period of almost 25 years. The newly introduced MO fully informed particle swarm (MOFIPS) optimization algorithm is found to provide better performing models with respect to those developed using the standard PSO, as well as advanced gradient-based optimization techniques. These findings encourage the use of an MO approach to NNRF cross-validated training with swarm optimization.

Key words | FIPS, multi-objective, neural network river forecasting, NNRF, particle swarm optimization, PSO

Riccardo Taormina
Kwok-wing Chau (corresponding author)
Department of Civil and Environmental Engineering,
Hong Kong Polytechnic University,
Hung Hom,
Kowloon,
Hong Kong,
China
E-mail: *cekwchau@polyu.edu.hk*

INTRODUCTION

Neural network river forecasting

Artificial neural networks (ANNs) have been successfully employed as black-box modeling tools in many hydrological contexts, mostly due to their intrinsic non-linear properties and adaptive behavior that grant them the ability to cope with the non-linear processes usually found in hydrology. Common examples range from rainfall forecasting (Lin & Wu 2009; Wu *et al.* 2010), to groundwater modeling (Coulibaly *et al.* 2000; Adamowski & Chan 2011; Taormina *et al.* 2012), water resources management (Adamowksi & Karapataki 2010) and water quality modeling (Banerjee *et al.* 2011), although most applications deal with rainfall-runoff modeling and streamflow forecasting (De Vos & Rientjes 2008; Islam 2010; Shamseldin 2010; Wu & Chau 2011). The latter applications have been collectively termed neural network river forecasting (NNRF) in a recent paper by many prominent authors of the field (Abrahart *et al.* 2012), and this work positions itself within this area of research. NNRF models are known to perform favorably against conceptual models (Tokar & Markus 2000; Carcano *et al.* 2008; Nayak *et al.* 2013), and although the latter enjoy physical interpretation, studies have shown that it is possible to identify hydrological sub-processes within a trained ANN architecture (Wilby *et al.* 2003; Jain *et al.* 2004; Jain & Kumar 2009). Different ANN architectures have found use among hydrologists. In particular, feed-forward neural networks (FNNs) have been largely employed due to their proven performances, consolidated training methods and universal approximation capability (Maier & Dandy 2000). When carrying out the

doi: 10.2166/hydro.2014.116

training process to map the unknown input–output relationship into the FNN architecture, the preferred techniques are first and second-order local search methods (Maier *et al.* 2010), such as the conjugate gradient (CG) and the Levenberg–Marquardt (LM) method.

Particle swarm optimization for NNRF model development

Although extremely fast, even the most advanced local search methods are prone to being trapped in local minima, especially in complex problems with a rough error surface and many local optima. Valid alternatives to the local search approach are represented by global search schema such as genetic algorithms (Chen & Chang 2009; Zhang *et al.* 2009), and the more recent particle swarm optimization (PSO) method (Eberhart & Kennedy 1995; Kennedy 2006; Poli *et al.* 2007). Although slower than local search methods, global search techniques can escape local minima. When employed for training neural networks, global optimization methods perform a parallel multipoint search on the error surface in which each point represents a hypothesis on the configuration of the network parameters. The degree of fitness of each candidate solution is computed based on the network predicted output, and a new generation of the population is created according to different updating rules. In the PSO algorithm, the population is made up by particles that behave like a storm of birds collectively looking for prey. In an iterative fashion, the position of each particle changes with a certain velocity, which is a function of the best position reached so far by the particle as well as the best position found among all the particles in its neighborhood. In recent times, the PSO algorithm has gained popularity in the field of hydrology due to its simplicity and computational efficiency, and successful applications are already available in the literature. However, most applications deal with the optimization of conceptual hydrological models (Gill *et al.* 2006; Scheerlinck *et al.* 2009; Zhang *et al.* 2009) and decision making for water resource management (Reddy & Kumar 2009; Montalvo *et al.* 2010; Guo *et al.* 2012). Only a few applications concerning the use of swarm optimization for NNRF applications are available in the literature, and results are limited and somewhat contradictory. Chau (2007, 2006) employed PSO-trained FNNs for real-time prediction of the water stage in the Shing Mun River, Hong Kong. The results showed that PSO-trained networks were more accurate than those optimized using common back-propagation or the LM algorithm. In addition, the author proposed the combination of PSO and LM in a split step approach (Chau 2007). This paradigm combines the advantages of global search capability of PSO algorithm in the first step, and local fast convergence of the LM algorithm in the second step. The mixed approach was able to attain a higher accuracy than the two algorithms on their own. Although these initial studies suggested that the PSO is able to perform better model calibration compared to derivative-based techniques, the opposite conclusions were drawn by Piotrowski & Napiorkowski (2011) for a recent case study involving streamflow discharge forecasting in the Annapolis River catchment, Nova Scotia (Canada). In their work, the authors show that LM training yielded more accurate FNN models in much less time with respect to several global optimization techniques, including two advanced variants of the PSO which are known to outperform the standard version of the algorithm on benchmark tests. The authors therefore strongly advocate for the use of local search techniques to develop NNRF models by means of differentiable objective functions, as long as a multi-start approach is implemented to reduce the chances of getting stuck in poor minima.

Objective of this study

Due to the limited number of reported applications, this present study is an attempt to shed light on the real effectiveness of swarm optimization techniques as a calibration algorithm for NNRF models. In particular, we focus on how generalization is ensured in PSO-trained neural networks, an aspect which has been overlooked so far and that could possibly explain the poorer performances of swarm optimization in some applications. Generalization is the ability of an ANN model to produce accurate approximations of the output variable given inputs that were not included in the training dataset. When gradient-based search methods are used to calibrate the ANN, model generalization is usually ensured through early stopping (ES) (Coulibaly *et al.* 2000). ES entails the division of the training

dataset in two subsets to form an independent validation dataset. The calibration is performed by minimizing the objective function on the training dataset, but it is stopped once the performances in the validation dataset start to deteriorate, a signal that the model is now fitting the random noise in the data rather than the underlying relationship between the variables. A similar split-set approach is employed to prevent over-fitting in PSO-trained NNRF models. Specifically, cross-validation is implemented by allowing particle position updates only if fitness improvements are concurrently recorded on both the training and the validation dataset (Piotrowski & Napiorkowski 2011), in analogy with stopped training. We suggest that this approach is wrong in two respects. In the first place, it is entirely possible for a particle to move to a location characterized by better generalization while temporarily underperforming on the validation dataset. This could be easily seen by running the PSO to minimize only the training errors while recording the evolution of the validation performances at the same time, as done in Figure 1 for a given particle in a trial experiment. The image shows how the validation error decreases in the long run although it goes up several times during the optimization process. Therefore, it is likely that preventing those trajectories resulting in temporary deterioration of validation performances might severely hinder the optimization process, especially in the early stages. Most importantly, unlike stopped training there is no effective difference in how the training and the validation datasets are used in PSO-ANN calibration. Indeed, since the acceptance rule for particle position update employs the objective functions defined on both datasets, the implementation of cross-validated training should be addressed as a MO optimization problem. This is the object of the present paper, in which a novel MO variant of the swarm optimization algorithm is employed for the development of NNRF models. The proposed algorithm MO fully informed particle swarm (MOFIPS) is a Pareto-based extension of the single-objective (SO) fully informed particle swarm (FIPS) optimization technique (Mendes *et al.* 2004). The MOFIPS is first shown to outperform other evolutionary-based MO techniques on benchmark tests, and then used to train NNRF models for the prediction of future streamflow discharges in the North Fork of the Shenandoah River, Virginia (USA). The MOFIPS is found to provide better performing models with respect to those developed using the standard PSO, as well as four advanced gradient-based optimization techniques, namely three variants of the CG method and the LM algorithm. A comparative analysis on different regular topologies shows that the effectiveness of the MOFIPS is subjected to how particles are arranged within the swarm. This work suggests that MO training methods could substantially improve the generalization ability of NNRF models through cross-validation, encouraging further research in this direction.

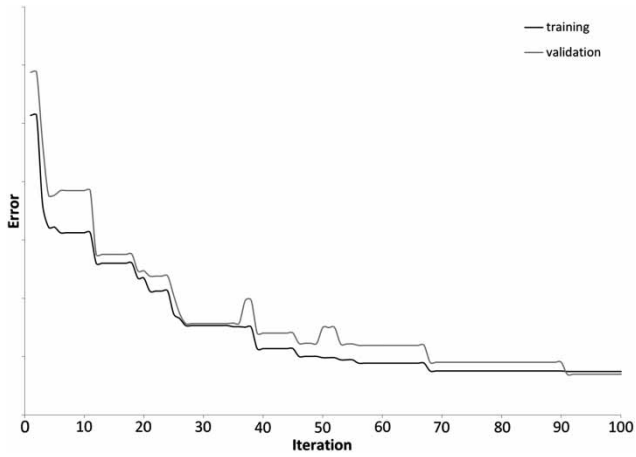

Figure 1 | Training and validation errors in non-cross-validated PSO-ANN training.

METHODS

Particle swarm optimization

In recent times, the PSO method (Eberhart & Kennedy 1995; Kennedy 2006; Poli *et al.* 2007) has gained popularity in many fields of science and engineering due to its computational efficiency, and relative simplicity of implementation compared to other global optimization approaches. When the PSO is employed for ANN training each particle represents a different hypothesis on the optimal set of parameters of the network, and consists of an n-dimensional vector where n is the number of parameters in the model. In the standard form of the PSO, the search is performed by updating the current position $\boldsymbol{X}_t$ of each particle in the n-dimensional parameter space with a velocity $\boldsymbol{V}_t$ that in turns depends on the particle best position as well as the

best position among all the particles in its neighborhood up to time t–1. The best positions of the particles are expressed in terms of highest fitness, or minimum value of the employed objective function which for ANN training is usually chosen as a function of the squared errors between the ANN approximations on a given dataset. If $\boldsymbol{P_i}$ is the best position for ith particle up to time t–1, and $\boldsymbol{G}$ is the global best position in the particle's neighborhood, the canonical PSO algorithm in its Type 1 constriction formulation (Clerc & Kennedy 2002) as:

$$\boldsymbol{V_t}=\chi(\boldsymbol{V_{t-1}}+\boldsymbol{U}[\boldsymbol{0},\boldsymbol{\varphi_1}]\otimes(\boldsymbol{P_i}-\boldsymbol{X_{t-1}})+\boldsymbol{U}[\boldsymbol{0},\boldsymbol{\varphi_2}]\otimes(\boldsymbol{G}-\boldsymbol{X_{t-1}})) \tag{1}$$

$$\boldsymbol{X_t}=\boldsymbol{X_{t-1}}+\boldsymbol{V_t} \tag{2}$$

where $\otimes$ is the point-wise vector multiplication; $\boldsymbol{U}[0,\varphi_i]$, $\boldsymbol{i}=1,2$ is a n-dimensional vector of uniformly distributed random numbers between 0 and φ_i; χ is a constriction coefficient that ensures convergence and is given by $\chi=\left[2/\left(\varphi-2+\sqrt{\varphi^2-4\varphi}\right)\right]$, where $\varphi=\varphi_1+\varphi_2>4$. A common choice that guarantees convergence is to set $\varphi_1=\varphi_2=2.05$, so that $\varphi=4.1$ and $\chi=0.7298$. The particle position update in Equation (2) is accepted only if the new location is characterized by a higher fitness value, or in the case of cross-validated ANN training, if an improvement of the performances on the training dataset is not associated with a reduction of its generalization performances on the validation dataset (Piotrowski & Napiorkowski 2011).

Fully informed PSO

The canonical version of the PSO entails that velocity updates are performed by considering only the position of the best particle in the neighborhood and a particle's best position. However, it has been suggested that considering the positions of the other particles in the neighborhood may benefit the search process carried out by the swarm as a whole (Mendes *et al.* 2003). This is the idea behind the FIPS variant of the PSO in which the best position of all the particle neighbors is taken into account for the velocity update (Mendes *et al.* 2004). The equations for the FIPS can be obtained from the Type 1' constriction formulation of the PSO by first noting that (2) is equivalent to:

$$\boldsymbol{V_t}=\chi(\boldsymbol{V_{t-1}}+\boldsymbol{\varphi}(\boldsymbol{P_m}-\boldsymbol{X_{t-1}})) \tag{3}$$

for $\varphi_{\max}=4.1$ and

$$\boldsymbol{\varphi_1}=\boldsymbol{U}\left(0,\frac{\varphi_{\max}}{2}\right) \tag{4a}$$

$$\boldsymbol{\varphi_2}=\boldsymbol{U}\left(0,\frac{\varphi_{\max}}{2}\right) \tag{4b}$$

$$\boldsymbol{\varphi}=\boldsymbol{\varphi_1}+\boldsymbol{\varphi_2} \tag{4c}$$

$$\boldsymbol{P_m}=\frac{\boldsymbol{\varphi_1}\otimes\boldsymbol{P_i}+\boldsymbol{\varphi_2}\otimes\boldsymbol{G}}{\boldsymbol{\varphi}} \tag{4d}$$

The FIPS formulation is obtained by modifying the partitioning in (4a)–(4d) to obtain:

$$\boldsymbol{\varphi_k}=\boldsymbol{U}\left(0,\frac{\varphi_{\max}}{|N|}\right)\ \forall k\in N \tag{5a}$$

$$\boldsymbol{\varphi}=\sum_{\boldsymbol{k}\in\boldsymbol{N}}\boldsymbol{\varphi_k} \tag{5b}$$

$$\boldsymbol{P_m}=\frac{\sum_{\boldsymbol{k}\in\boldsymbol{N}}W(k)\boldsymbol{\varphi_k}\otimes\boldsymbol{P_k}}{\sum_{\boldsymbol{k}\in\boldsymbol{N}}W(k)\boldsymbol{\varphi_k}} \tag{5c}$$

where $\boldsymbol{P_k}$ is the best position found by the kth neighbor of the particle to be updated, and $W(k)$ is a weighting factor which is a constant value or a function of some relevant characteristic of the kth particle, such as its fitness or its Euclidean distance from the particle being updated.

Swarm topology

The swarm topology is defined as the layout of particles connections forming the neighborhoods. Although dynamic topologies with changing neighborhoods have been employed, for the remainder of the discussion they will be considered fixed during the optimization process, unless otherwise specified. The first topology employed in the

PSO algorithm is the ball of Figure 2(a), a fully connected graph where each particle neighborhood comprises all the other particles in the swarm. Other topologies were later introduced (Kennedy 1999; Kennedy & Mendes 2002; Poli *et al.* 2007) such as the ring (Figure 2(b)), lattices (Figure 2(c)), or clustered geometries (Figure 2(d)), as well as custom topologies and random graphs. Each topology provides a different balance between the exploration and exploitation phase of the search, and their performances may vary greatly depending on the optimization problem at hand. The same topology is found to affect the search process in a very different way depending on which version of the swarm optimization algorithm is being employed. For instance, the ball topology hastens convergence and penalizes variety in canonical PSO since one global best is selected for all the particles in the swarm. On the other hand, the same arrangement greatly promotes diversity in the FIPS since every particle contributes to the movement of each other particle in the swarm. In addition, while the particle's own position is always employed in the PSO, FIPS topologies may or may not include it.

The MOFIPS algorithm

In MO optimization (MO) problems, several conflicting objective functions have to be minimized concurrently (Deb 2009). In this study, we address cross-validated PSO-ANN training as a bi-objective optimization problem in which the two objective functions to minimize are the mean-squared-errors (MSEs) of the ANN approximations on the training (MSE_T) and validation dataset (MSE_V), respectively. The two MSEs are conflicting since an improvement of one statistic due to over-fitting will result in a loss of performance on the other dataset. The most common strategy employed to solve MO problems is that of Pareto-optimality (Gill *et al.* 2006; Reddy & Kumar 2009). Due to the existence of multiple objective functions, the final outcome returned by a Pareto-based algorithm is

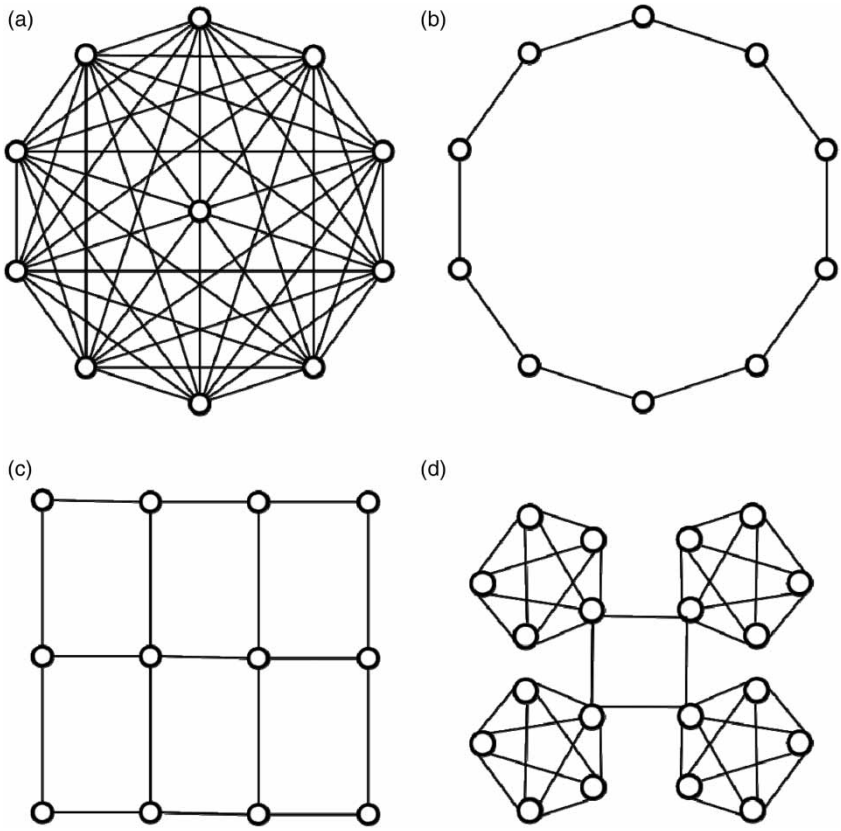

Figure 2 | Examples of swarm topologies: (a) ball; (b) ring; (c) lattice; (d) cluster.

not a unique solution, but rather a set of equally good candidates presenting different trade-offs with respect to the objectives. These solutions are said to be non-dominated or Pareto-efficient, meaning that there is no other candidate showing simultaneously a higher fitness value in each of the objective functions to minimize. When plotting the values of the objective functions against each other, the set of non-dominated solutions describe a frontier known as the Pareto-front, which is displayed in Figure 3 for the cross-validated PSO-ANN training case. A successful algorithm should return a set of solutions which are close to the real Pareto-front and equally distributed along the frontier. MO generalizations of the PSO (MOPSO) algorithm usually implement the Pareto-based approach by maintaining a set of non-dominated particle positions with respect to the swarm, known as leading or guiding particles. The positions of these particles are stored separately for reference, and the archive is updated constantly by including new non-dominated solutions and excluding those which end up being dominated at each optimization step. When the archive grows too big, clustering and trimming operations are carried out to retain those representative solutions offering the most uniform spread along the Pareto-front. The other particles in the swarm can access the archive and direct their search by picking up sensible leaders through a variety of selection schemes (Reyes-Sierra & Coello Coello 2006). In this study, we introduce a MOFIPS based on Pareto-dominance, which represents an extension of the single objective FIPS algorithm described in earlier paragraphs. The leading particles forming the Pareto-front are not stored separately in an archive but they are added to the original swarm topology, which is now allowed to change. This is shown in Figure 4 for a swarm with four particles and one leader $\boldsymbol{L}$. The directed connections originating from $\boldsymbol{L}$ indicate that the particles in the Pareto-front are not subjected to the influence of the other particles, therefore they will not move during the optimization step and their velocity is set to zero. Leading particles are instances of the non-dominated positions found during the search, acting as guides or centers of attraction for the entire swarm. The Pareto-front is updated at each iteration by including new non-dominated positions until a maximum size M of the front is reached. After this maximum is reached, the update will be carried out after taking into account the crowding distance associated with each non-dominated position. The crowding distance is a measure of the density of non-dominated positions in a certain area of the objective function space. It has been firstly introduced as part of the non-dominated sorting genetic algorithm II (NSGA-II) to foster diversity among possible solutions (Deb *et al.* 2002). The MOFIPS promotes diversity in the Pareto-optimal set by retaining only the first M positions with the highest crowding distances and discarding the others. To improve swarm convergence and prevent local minima entrapment a turbulence factor is introduced in the form of a polynomial mutation operator (Deb 2009). The MOFIPS algorithm thus described requires the specification of three more parameters with respect to the single objective FIPS. These are the maximum number M of particles in the Pareto-front, as well as the percentage μ of

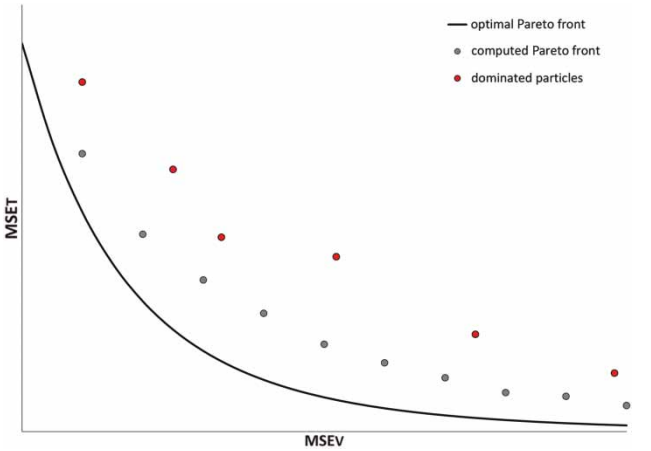

Figure 3 | Pareto-front for the PSO-ANN training case.

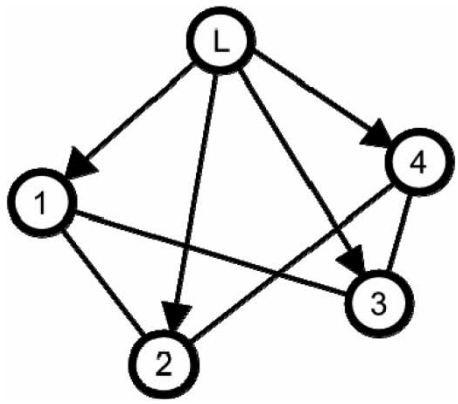

Figure 4 | MOFIPS topology with four particles and one leader (*L*).

particles dimensions subjected to turbulence, and the parameter η controlling the probability distribution of the polynomial operator. Preliminary trials not reported here have shown that simplifying the FIPS equations results in better MOFIPS performances. In particular, the abstract position $\boldsymbol{P_m}$ in Equation (5c) is now computed as a basic weighted sum of its neighbors' best positions (fixed neighbors plus leading particles). In particular, each dimension of the kth neighbor best position $\boldsymbol{P_k}$ is multiplied by the same weight $\left[\varphi_k / \sum_{k \in N} \varphi_k\right]$, where φ_k is a uniform random number between 0 and 1. The MOFIPS formulation can be thus expressed as:

$$\boldsymbol{V_t} = \chi(\boldsymbol{V_{t-1}} + \varphi_{\max}(\boldsymbol{P_m} - \boldsymbol{X_{t-1}})) \tag{6a}$$

$$\boldsymbol{P_m} = \sum_{k \in N} \left(\frac{\varphi_k}{\sum_{k \in N} \varphi_k} \boldsymbol{P_k} \right) \tag{6b}$$

$$\boldsymbol{X_t} = \boldsymbol{X_{t-1}} + \boldsymbol{V_t} \tag{6c}$$

MOFIPS performances on benchmark tests

MOFIPS performances were assessed on the experiments proposed in Deb *et al.* (2002) for testing the NSGA-II algorithm, one of the most widely used MO evolutionary algorithms (MOEAs) these days. In their work, the authors compare the real and binary-coded version of the NSGA-II against two other common MOEAs, namely the Pareto-archived evolution strategy (PAES) (Knowles & Corne 1999) and the strength Pareto-evolutionary algorithm (SPEA) (Zitzler & Thiele 1999). The comparison is carried out on a set of both unconstrained and constrained test problems for which the Pareto-optimal set is known. For the purpose of this study, only MOFIPS performances for unconstrained optimization were considered. In particular, the MOFIPS algorithm was tested for the unconstrained version of the SCH, FON, ZDT1, ZDT2, ZDT3 and ZDT6 problems. The performances were estimated using the same two performance measures employed in Deb *et al.* (2002), which are particularly effective in directly evaluating both the convergence to a known Pareto-optimal set and the spread among the solutions returned by the algorithm. These two measures are, respectively, called the convergence metric Υ and the diversity metric Δ. A value of zero of Υ entails perfect convergence of the algorithm solutions to a chosen subset of points in the optimal Pareto-front. Accordingly, a zero value for the diversity metric Δ will identify that a set of solutions spans uniformly the entire Pareto-front, including the extremes. The reader is referred to the original paper for information on how these two metrics are actually computed. For fair comparison with the results reported in the study, the same maximum of 25,000 function evaluations was set as the termination criterion for each MOFIPS simulation run. In Tables 1 and 2, the mean and variance of the two performance measures are reported for NSGA-II, SPEA and PAES as featured in the work by Deb *et al.* (2002). The statistics for a MOFIPS configuration are also displayed for comparisons. This configuration represents a ring-structured swarm with 20 particles, two neighbors per particle, a maximum size of the Pareto-front M of 20 particles, with turbulence

Table 1 | Mean (first row) and variance (second row) of the convergence metric Υ (in bold text the lowest value recorded for each test problem)

	SCH	FON	ZDT1	ZDT2	ZDT3	ZDT6
NSGA-II real-coded	0.003391	0.001931	0.033482	0.072391	0.114500	0.296564
	0.000000	0.000000	0.004750	0.031689	0.007940	0.013100
NSGA-II binary-coded	0.002833	0.002571	**0.000894**	**0.000824**	0.043411	7.806798
	0.000001	0.000000	0.000000	0.000000	0.000042	0.001667
SPEA	0.003403	0.125692	0.001799	0.001339	0.047517	0.221138
	0.000000	0.000038	0.000001	0.000000	0.000047	0.000449
PAES	**0.001313**	0.151263	0.082085	1.126276	0.023872	0.085469
	0.000003	0.000905	0.008679	0.036877	0.000010	0.006664
MOFIPS	0.003167	**0.001055**	0.003840	0.002237	**0.003925**	**0.011751**
	0.000000	0.000000	0.000002	0.000001	0.000000	0.000818

Table 2 | Mean (first row) and variance (second row) of the diversity metric Δ (in bold text the lowest value recorded for each test problem)

	SCH	FON	ZDT1	ZDT2	ZDT3	ZDT6
NSGA-II real-coded	0.477899	0.378065	0.390307	0.430776	0.738540	0.668025
	0.003471	0.000639	0.001876	0.004721	0.019706	0.009923
NSGA-II binary-coded	0.449265	0.395131	0.463292	0.435112	0.575606	0.644477
	0.002062	0.001314	0.041622	0.024607	0.005078	0.035042
SPEA	1.021110	0.792352	0.784525	0.755148	0.672938	0.849389
	0.004372	0.005546	0.004440	0.004521	0.003587	0.002713
PAES	1.063288	1.162528	1.229794	1.165942	0.789920	1.153052
	0.002868	0.008945	0.004839	0.007682	0.001653	0.003916
MOFIPS	**0.254751**	**0.373575**	**0.382408**	**0.375675**	**0.498458**	**0.432655**
	0.003146	0.006032	0.004072	0.004741	0.002196	0.050052

parameters μ and η of 0.2 and 20, respectively. From the analysis of the results in the tables, it can be seen that MOFIPS compares extremely well against all the other MOEAs. The MOFIPS consistently outperforms every other algorithm in terms of the diversity metric Δ for each of the benchmark problems. Significant improvements are not only recorded against the less performing PAES and SPEA, but also with respect to both versions of the NSGA-II. Results are also very promising when the convergence metric Υ is considered. The MOFIPS shows best convergence in half of the benchmarks problems (FON, ZDT3 and ZDT6), with PAES coming first once (SCH), and the binary version of the NSGA-II twice (ZDT1 and ZDT2). In addition, the MOFIPS appears to be the most balanced among all the algorithms, showing good convergence for all benchmarks.

STUDY AREA

The effectiveness of the MOFIPS for developing cross-validated NNRF models will be tested against SO swarm optimization algorithms and gradient-based techniques on a streamflow forecasting application in the Shenandoah River watershed. The Shenandoah River is a flood-prone river in Virginia, USA, and is the principal tributary of the Potomac River. The Shenandoah River is originated by the confluence of two tributaries, the South Fork and the North Fork, which join their courses northeast of the city of Front Royal in Warren County (Figure 5). In this work, we are concerned with 1-day ahead forecasting of river discharge in the North Fork of the Shenandoah River, a fifth-order stream of 169 km that drains an area of around 6,930 km^2 of north eastern Virginia. Daily river discharge observations are available from a gauging station in Strasburg, while daily precipitation data are collected from a meteorological station in Waterloo sited around 35 km upstream from the gauging station. Around 9,000 observations of river discharge and rainfall were retrieved from the US Geological Survey database, ranging from May 1985 to the end of December 2009. A sample of the recorded times series is given in Figure 6.

RESULTS AND DISCUSSION

Input and model selection

The original time series were initially pre-processed to remove outliers and additional inputs were formed by means of aggregating operators. In particular, 3-day and 7-day moving averages of flow observations, as well as 3-day and 7-day cumulated precipitation, were employed to form a total of six input variables. Lagged time series up to 3 days were considered so that the total set of inputs comprised a total of 18 potential candidates. The input and output variables were rescaled in the [–1, 1] range to facilitate ANN training, and the dataset was then split into a training (40% of available dates), a validation (40% of available dates) and a test dataset (20% of available dates). A constructive forward selection (CFS)

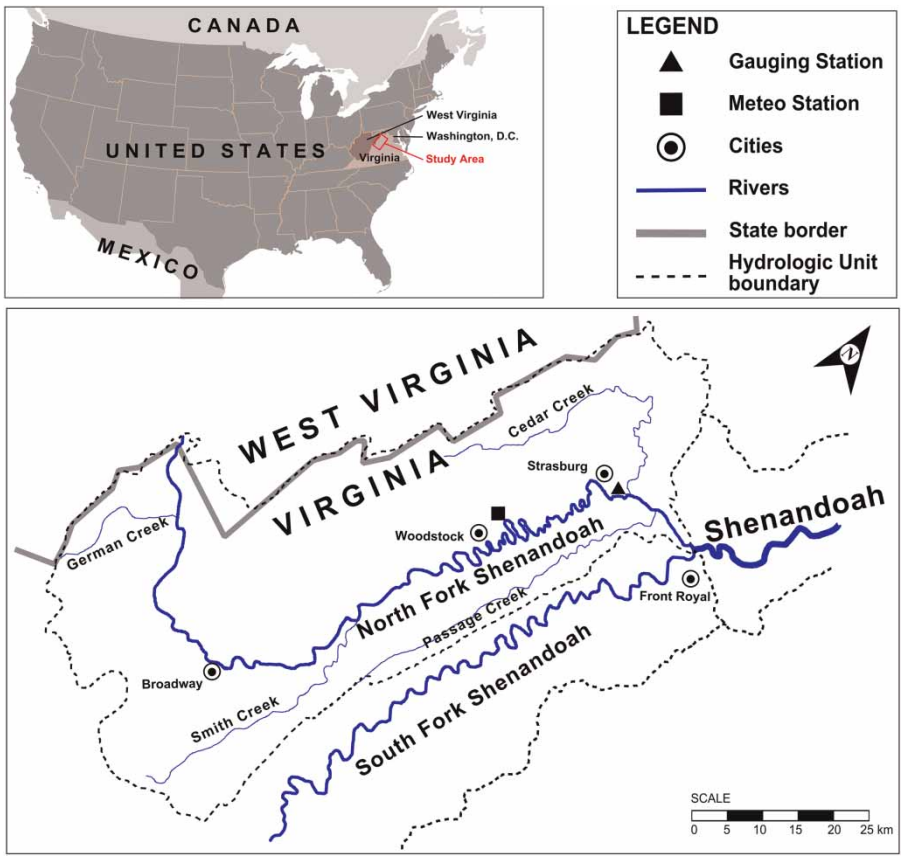

Figure 5 | Location of the Shenandoah River.

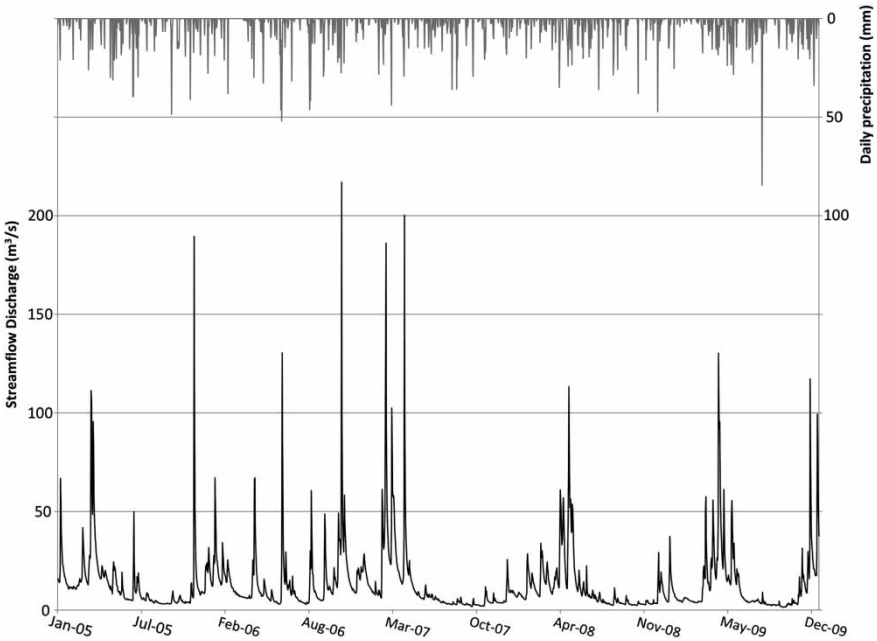

Figure 6 | Sample of recorded total precipitation and streamflow discharge.

scheme (Maier *et al.* 2010) was then implemented to select the optimal number of hidden neurons of the FNN, as well as the optimal combination of inputs among all candidates. The CFS entails an incremental trial-and-error strategy where an initial ANN with minimal complexity is trained n times separately, each time having only one of the n candidate variables as the sole input. The most significant input is chosen according to an optimality criterion, and the search continues by looking for the next input among the remaining n–1 candidates to add to the model. This procedure is repeated iteratively until the inclusion of further inputs does not yield any improvement of the optimality criterion. When this event occurs, the ANN is first augmented with an additional neuron to its hidden layer, and the search for new inputs is resumed from where it had stopped. The search continues if adding a new input to the augmented model results in improved performances, otherwise the CFS process is terminated and the model obtained at the previous step is returned. To improve the reliability of the CFS scheme, the optimality criterion in this study was chosen as the median value of the validation MSEs obtained by training each ANN model with 100 restarts. Due to the computational effort needed to perform the CFS, the ANNs were trained using the LM method, which is the fastest training technique employed in this study. The optimal model returned by the CFS scheme was found to have six hidden neurons and five input variables, namely the streamflow discharges up to 3 days ahead (t–1, t–2 and t–3), the rainfall measured the previous day (t–1), and the 3-day cumulated rainfall computed at time t–3. The total number of weights (including ANN biases) in the optimal model was 43.

Comparison of PSO and MOFIPS algorithms' performances

The optimal model returned by the CFS scheme was trained with SO PSO as well as the MOFIPS algorithm to check whether addressing cross-validated ANN training as a MO problem would result in improved performances. The comparison was carried out using the four particle arrangements in Figures 2(a)–2(d) so to assess the impact of different topologies on the quality of the developed NNRF model. All the employed topologies were made of 30 particles that were disposed to form a ball of 30 neighbors, a ring of 30 particles with two neighbors each, a 6-by-5 bi-dimensional lattice, and a clustered arrangement with five niches of six particles each. For the MOFIPS case, these topologies were tested with or without including each particle in its own neighborhood. These cases will be respectively identified as MOFIPS-w and MOFIPS-wo for the remainder of the discussion. The maximum number M of Pareto-front particles as well as the parameter η controlling the turbulence probability distribution were set to 30, while the percentage of particles subjected to turbulence was set to $\mu = (1/\text{number of weights}) \cong 0.023$. Each PSO/MOFIPS case was run a total of 20 times for 1,000 iterations, and a distribution of the performances on each dataset was obtained by extracting the best performing particle of each run. The best performing solution for each PSO run was obtained by selecting the particle with the best validation performances. On the other hand, a strategy had to be developed to select a solution from the Pareto-front returned by each MOFIPS run. Each solution in the Pareto-front is by definition equally efficient with respect to the objectives; however, the solutions at extremes of the frontier are more likely to over-fit one of the two datasets. Accordingly, the solutions located around the knee of the Pareto-front should theoretically have better generalization performances, especially if the algorithm is able to return a dense and equally spaced set of solutions. Although more sophisticated selection schemes could be implemented, in this work, the optimal solution of each MOFIPS run was chosen as the one located at the knee of the frontier. The statistical comparison of PSO and MOFIPS performances is reported in Table 3 for each employed topology in terms of the Nash–Sutcliffe coefficient of efficiency (COE). Other goodness-of-fit measures were employed for the comparison, but these results were not reported since they will not add further insights to the analysis. From a first glance at Table 3, it appears that the NNRF models trained with the MOFIPS algorithm clearly outperform than those obtained with standard PSO. With the exception of one topology, the median values of the COEs are around 5% higher on the training and validation datasets, and over 11% higher on the test dataset. These figures suggest that the MO approach to PSO-ANN training

Table 3 | Comparison of PSO and MOFIPS for cross-validated ANN training

	Training					Validation					Test				
	COE					COE					COE				
	Best	Worst	Median	Mean	STD	Best	Worst	Median	Mean	STD	Best	Worst	Median	Mean	STD
PSO															
Ball	0.791	0.666	0.757	0.750	0.033	0.788	0.663	0.777	0.767	0.032	0.722	0.432	0.663	0.649	0.068
Ring	0.768	0.633	0.728	0.725	0.034	0.787	0.648	0.741	0.737	0.032	0.697	0.432	0.615	0.610	0.066
Lattice	0.785	0.704	0.746	0.744	0.021	0.784	0.724	0.753	0.753	0.018	0.714	0.531	0.637	0.629	0.045
Clustered	0.778	0.696	0.749	0.746	0.023	0.793	0.697	0.764	0.758	0.025	0.719	0.526	0.646	0.633	0.047
MOFIPS-w															
Ball	0.771	− 0.025	0.753	0.714	0.174	0.772	− 0.001	0.757	0.719	0.170	0.678	− 0.008	0.634	0.604	0.145
Ring	0.805	0.754	0.784	0.783	0.012	0.792	0.765	0.782	0.781	0.008	0.741	0.662	0.704	0.704	0.021
Lattice	0.802	− 0.029	0.783	0.743	0.182	0.795	− 0.001	0.782	0.743	0.175	0.734	− 0.004	0.708	0.670	0.160
Clustered	0.795	0.761	0.784	0.781	0.011	0.793	0.768	0.782	0.782	0.006	0.730	0.655	0.697	0.696	0.021
MOFIPS-wo															
Ball	0.773	− 0.022	0.753	0.716	0.174	0.779	− 0.003	0.755	0.719	0.170	0.678	− 0.004	0.637	0.607	0.146
Ring	0.790	0.748	0.777	0.775	0.011	0.794	0.754	0.780	0.779	0.010	0.732	0.629	0.686	0.688	0.027
Lattice	0.802	0.752	0.787	0.782	0.015	0.793	0.757	0.784	0.779	0.010	0.732	0.609	0.717	0.702	0.033
Clustered	0.803	0.762	0.786	0.784	0.012	0.798	0.760	0.780	0.781	0.009	0.731	0.645	0.710	0.699	0.028

introduced in this work provides more successful NNRF models. The analysis of the results for each topology shows that the efficiency of both PSO and MOFIPS depends on the adopted swarm arrangement. In particular, the PSO seems to be more efficient when the ball topology is employed. This could be attributed to the higher convergence speed provided by this arrangement under the PSO framework with respect to the other topologies. On the other hand, the increased flow of information among neighbors penalizes the ball topology for both the MOFIPS cases. These results are consistent with those obtained by comparing the PSO and FIPS algorithms on benchmark trials (Mendes *et al.* 2004), although the MOFIPS employs an alternative formulation of the fully informed velocity update (Equation (6b)). All the other candidate topologies are more or less successful and there are no significant differences between the MOFIPS-w and MOFIPS-wo cases. However, the MOFIPS-wo lattice combination is arguably the one providing the best results overall, with the MOFIPS-wo clustered geometry coming a close second.

Comparison of MOFIPS and gradient-based algorithms' performances

After showing the superiority of MOFIPS over standard PSO for cross-validated training, an experiment was carried out to assess how the proposed MO approach would compare against four of the most advanced local search algorithms employed for NNRF development and ANN training (Hamed *et al.* 2004; Chen & Chang 2009; Adamowski & Karapataki 2010). These algorithms are the scaled CG (SCG), the CG with Fletcher–Reeves (CGF) updates, the CG with Polak–Ribiére (CGP) updates, and the LM method. All these algorithms are included in the Neural Network Toolbox of the Matlab® computing suite, which has also been employed to implement the swarm optimization algorithms of this work. All the local search methods were run with 100 restarts in order to prevent poor minima entrapment, and the algorithm parameters were set as suggested by Matlab®. In Table 4, the NNRF performances for the prediction of the Shenandoah River discharges are reported in terms of COE, root MSE (RMSE), and the ratio of the RMSE to the standard deviation (RMSE/STDEV). The values shown are for the best model in terms of validation performances selected among those returned by each algorithm after 100 restarts. The second column of Table 4 reports the computational time needed by each algorithm to perform all the restarts. It can be seen that the model obtained with the LM algorithm outperforms all those obtained with the other local search methods on each dataset, except for the CGF-trained ANN that shows basically the same performances on the test dataset. Nonetheless, while the LM is the fastest algorithm, the CGF is the slowest and it needs a much longer computational time to provide the model with this level of accuracy. The superiority of the LM over CG methods was also reported in other studies (Hamed *et al.* 2004; Adamowski & Karapataki 2010). The last row of Table 4 shows the performances of a MOFIPS run for comparison with the gradient-based techniques. The results were obtained with the MOFIPS-wo lattice configuration, this time after running 4,000 iterations to ensure convergence. The optimal MOFIPS solution was selected according to the scheme presented in the previous paragraph. From the analysis of the results, it emerges that

Table 4 | Comparison of PSO and gradient-based techniques for cross-validated ANN training

		RMSE (m^3/s)			RMSE/STDEV			COE		
Training algorithm	**Time (s)**	**Training**	**Validation**	**Test**	**Training**	**Validation**	**Test**	**Training**	**Validation**	**Test**
LM	541	12.283	12.917	9.091	0.418	0.475	0.520	0.810	0.800	0.729
SCG	2,344	12.919	13.260	9.316	0.440	0.488	0.533	0.790	0.789	0.716
CGP	2,387	14.233	13.789	9.989	0.484	0.507	0.572	0.745	0.772	0.673
CGF	3,075	12.865	13.138	9.145	0.438	0.483	0.523	0.792	0.793	0.726
MOFIPS-wo lattice	956	12.232	13.147	8.822	0.416	0.484	0.505	0.812	0.793	0.745

the MOFIPS algorithm compares well against the gradient-based methods both in terms of time and model accuracy. Although slower than the LM, the MOFIPS is able to provide the NNRF with best generalization ability in 1/3 to 1/2 of the time employed by the CG algorithms to perform 100 restarts. The improvements provided by the MOFIPS on the test dataset range from a minimum of 2% for the LM to a maximum of 10% for the CGP, which is the worst performing algorithm. More insights on the relative NNRF performances can be obtained by inspecting the hydrograph fittings graphically, as done in Figure 7 for a sample of the training dataset. It can be seen that the NNRF model obtained by the MOFIPS and the LM tend to better approximate the peaks in streamflow discharge due to rainfall. On the other hand, all the neural networks seem to perform equally well in predicting the falling limbs of the hydrographs after a storm event. The CGP-trained model seems to suffer from timing errors in predicting the streamflow peaks. This is more likely to happen when there is an excessive imbalance in the relative importance of past streamflow input features over rainfall ones, suggesting that 100 restarts were not sufficient for the CPG algorithm to escape poor minima.

SUMMARY AND CONCLUSIONS

In this study, we propose a MO approach for cross-validated swarm optimization training of ANNs to be used for streamflow forecasting purposes. We suggest that addressing cross-validated training as a SO problem may hinder the optimization process performed by the swarm as it penalizes the exploratory behavior of the algorithm. In addition, the specification of two different objective functions in the acceptance rule for particle position update renders the optimization problem intrinsically MO, thus it should be treated as such. We therefore introduce a MOFIPS and employ it to calibrate a NNRF model for a streamflow prediction application in the Shenandoah River watershed. After validating MOFIPS performances on benchmark tests, the algorithm is first tested against the standard PSO on the case study application. After assessing the superiority of the proposed MO approach over SO cross-validated training, the performances of the MOFIPS algorithm on the case study application are checked against those of four advanced gradient-based techniques in multi-start mode. The NNRF model produced by the MOFIPS algorithm is found to outperform those built with all the other gradient-based algorithms, including the LM method. With the

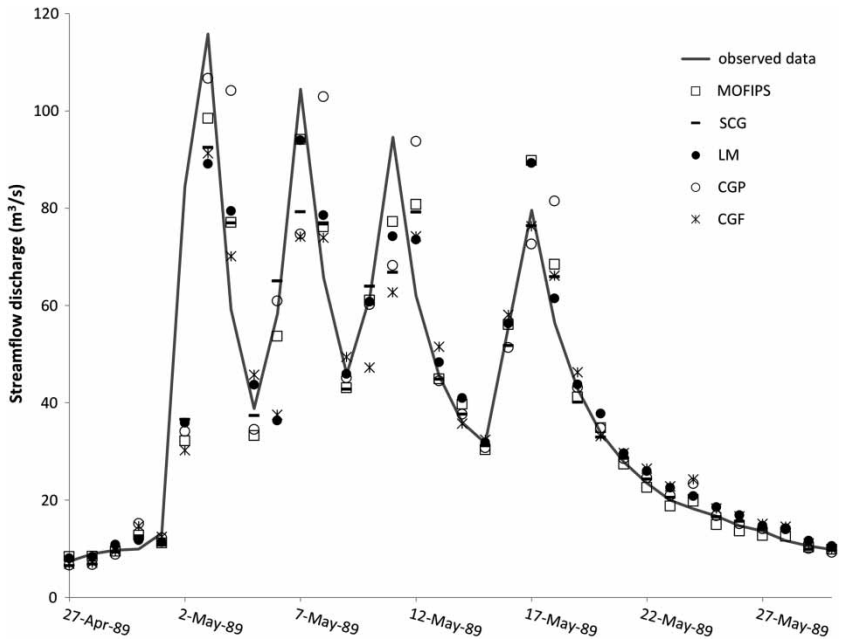

Figure 7 | Comparison of model predictions.

exclusion of the LM, such improvements are obtained with a significant reduction in the computational costs, especially in comparison to the slower methods of the CG family. Although further research is needed to thoroughly assess the effectiveness of the proposed MO training scheme for NNRF development, the findings of this work are certainly encouraging with respect to some results remarking the inferiority of SO swarm optimization (Piotrowski & Napiorkowski 2011). In this regard, it would be particularly interesting to check whether similar MO schemes can benefit NNRF training when other global optimization algorithms are used.

ACKNOWLEDGEMENTS

This research was funded by the Hong Kong PhD Fellowship Scheme established by the Research Grants Council (RGC) of Hong Kong and the Central Research Grant of Hong Kong Polytechnic University (G-U833).

REFERENCES

Abrahart, R. J., Anctil, F., Coulibaly, P., Dawson, C. W., Mount, N. J., See, L. M., Shamseldin, A. Y., Solomatine, D. P., Toth, E. & Wilby, R. L. 2012 Two decades of anarchy? Emerging themes and outstanding challenges for neural network river forecasting. *Prog. Phys. Geog.* **36** (4), 418–513.

Adamowski, J. & Chan, H. F. 2011 A wavelet neural network conjunction model for groundwater level forecasting. *J. Hydrol.* **407** (1–4), 28–40.

Adamowski, J. & Karapataki, C. 2010 Comparison of multivariate regression and artificial neural networks for peak urban water-demand forecasting: evaluation of different ANN learning algorithms. *J. Hydrol. Eng.* **15** (10), 729–743.

Banerjee, P., Singh, V. S., Chatttopadhyay, K., Chandra, P. C. & Singh, B. 2011 Artificial neural network model as a potential alternative for groundwater salinity forecasting. *J. Hydrol.* **398** (3–4), 212–220.

Carcano, E. C., Bartolini, P., Muselli, M. & Piroddi, L. 2008 Jordan recurrent neural network versus IHACRES in modelling daily streamflows. *J. Hydrol.* **362** (3–4), 291–307.

Chau, K. W. 2006 Particle swarm optimization training algorithm for ANNs in stage prediction of Shing Mun River. *J. Hydrol.* **329** (3–4), 363–367.

Chau, K. W. 2007 A split-step particle swarm optimization algorithm in river stage forecasting. *J. Hydrol.* **346** (3–4), 131–135.

Chen, Y. & Chang, F. 2009 Evolutionary artificial neural networks for hydrological systems forecasting. *J. Hydrol.* **367** (1–2), 125–137.

Clerc, M. & Kennedy, J. 2002 The particle swarm – explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.* **6** (1), 58–73.

Coulibaly, P., Anctil, F. & Bobée, B. 2000 Daily reservoir inflow forecasting using artificial neural networks with stopped training approach. *J. Hydrol.* **230** (3–4), 244–257.

De Vos, N. J. & Rientjes, T. H. M. 2008 Multiobjective training of artificial neural networks for rainfall-runoff modeling. *Water Resour. Res.* **44** (8), doi:10.1029/2007WR006734.

Deb, K. 2009 *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, New York.

Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T. 2002 A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **6** (2), 182–197.

Eberhart, R. & Kennedy, J. 1995 A new optimizer using particle swarm theory. Micro Machine and Human Science, 1995. MHS '95. In: *Proceedings of the Sixth International Symposium*, pp. 39–43.

Gill, M. K., Kaheil, Y. H., Khalil, A., McKee, M. & Bastidas, L. 2006 Multiobjective particle swarm optimization for parameter estimation in hydrology. *Water Resour. Res.* **42** (7), doi:10.1029/2005WR004528.

Guo, X., Hu, T., Zhang, T. & Lv, Y. 2012 Bilevel model for multi-reservoir operating policy in inter-basin water transfer-supply project. *J. Hydrol.* **424–425** (6), 252–263.

Hamed, M. M., Khalafallah, M. G. & Hassanien, E. A. 2004 Prediction of wastewater treatment plant performance using artificial neural networks. *Environ. Model. Softw.* **19** (10), 919–928.

Islam, A. 2010 Improving flood forecasting in Bangladesh using an artificial neural network. *J. Hydroinf.* **12** (3), 351–364.

Jain, A. & Kumar, S. 2009 Dissection of trained neural network hydrologic models for knowledge extraction. *Water Resour. Res.* **45** (7), doi:10.1029/2008WR007194.

Jain, A., Sudheer, K. & Srinivasulu, S. 2004 Identification of physical processes inherent in artificial neural network rainfall runoff models. *Hydrol. Process.* **18** (3), 571–581.

Kennedy, J. 1999 Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. Evolutionary Computation, 1999. CEC 99. *Proceedings of the 1999 Congress* **3**, 1931–1938.

Kennedy, J. 2006 Swarm intelligence. In: *Handbook of Nature-Inspired and Innovative Computing* (A. Zomaya, ed.). Springer, USA, pp. 187–219.

Kennedy, J. & Mendes, R. 2002 Population structure and particle swarm performance. Evolutionary Computation, 2002. CEC '02. *Proceedings of the 2002 Congress* **2**, 1671–1676.

Knowles, J. & Corne, D. 1999 The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation. Evolutionary Computation, 1999. CEC 99. *Proceedings of the 1999 Congress* **1**, 5 pp.

Lin, G. & Wu, M. 2009 A hybrid neural network model for typhoon-rainfall forecasting. *J. Hydrol.* **375** (3–4), 450–458.

Maier, H. R. & Dandy, G. C. 2000 Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications. *Environ. Model. Softw.* **15** (1), 101–124.

Maier, H. R., Jain, A., Dandy, G. C. & Sudheer, K. P. 2010 Methods used for the development of neural networks for the prediction of water resource variables in river systems: Current status and future directions. *Environ. Model. Softw.* **25** (8), 891–909.

Mendes, R., Kennedy, J. & Neves, J. 2003 Watch thy neighbor or how the swarm can learn from its environment. Swarm Intelligence Symposium, 2003. SIS '03. *Proceedings of the 2003 IEEE* **1**, 88–94.

Mendes, R., Kennedy, J. & Neves, J. 2004 The fully informed particle swarm: simpler, maybe better. *IEEE Trans. Evol. Comput.* **8** (3), 204–210

Montalvo, I., Izquierdo, J., Pérez-García, R. & Herrera, M. 2010 Improved performance of PSO with self-adaptive parameters for computing the optimal design of water supply systems. *Eng. Appl. Artif. Intell.* **23** (5), 727–735.

Nayak, P., Venkatesh, B., Krishna, B. & Jain, S. 2013 Rainfall runoff modelling using conceptual, data driven and wavelet based computing approach. *J. Hydrol.* **493** (17), 57–67.

Piotrowski, A. P. & Napiorkowski, J. J. 2011 Optimizing neural networks for river flow forecasting – evolutionary computation methods versus the Levenberg–Marquardt approach. *J. Hydrol.* **407** (1–4), 12–27.

Poli, R., Kennedy, J. & Blackwell, T. 2007 Particle swarm optimization. *Swarm Intell.* **1** (1), 33–57.

Reddy, M. J. & Kumar, D. N. 2009 Performance evaluation of elitist-mutated multi-objective particle swarm optimization for integrated water resources management. *J. Hydroinf.* **11** (1), 78–88.

Reyes-Sierra, M. & Coello Coello, C. A. 2006 Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. Technical Report EVOCINV-01-2006, Evolutionary Computation Group at CINVESTAV, Seccíon de Computacíon, Departamento de Ingeniería Eléctrica, CINVESTAV-IPN, Mexico.

Scheerlinck, K., Pauwels, V. R. N., Vernieuwe, H. & De Baets, B. 2009 Calibration of a water and energy balance model: Recursive parameter estimation versus particle swarm optimization. *Water Resour. Res.* **45** (10), doi:10.1029/2009WR008051.

Shamseldin, A. 2010 Artificial neural network model for river flow forecasting in a developing country. *J. Hydroinform.* **12** (1), 22–35.

Taormina, R., Chau, K. & Sethi, R. 2012 Artificial neural network simulation of hourly groundwater levels in a coastal aquifer system of the Venice lagoon. *Eng. Appl. Artif. Intell.* **25** (8), 1670–1676.

Tokar, A. S. & Markus, M. 2000 Precipitation-runoff modeling using artificial neural networks and conceptual models. *J. Hydrol. Eng.* **5** (2), 156–161.

Wilby, R., Abrahart, R. & Dawson, C. 2003 Detection of conceptual model rainfall–runoff processes inside an artificial neural network. *Hydrol. Sci. J.* **48** (2), 163–181.

Wu, C. L. & Chau, K. W. 2011 Rainfall–runoff modeling using artificial neural network coupled with singular spectrum analysis. *J. Hydrol.* **399** (3–4), 394–409.

Wu, C. L., Chau, K. W. & Fan, C. 2010 Prediction of rainfall time series using modular artificial neural networks coupled with data-preprocessing techniques. *J. Hydrol.* **389** (1–2), 146–167.

Zhang, X., Srinivasan, R., Zhao, K. & Liew, M. V. 2009 Evaluation of global optimization algorithms for parameter calibration of a computationally intensive hydrologic model. *Hydrol. Process.* **23** (3), 430–441.

Zitzler, E. & Thiele, L. 1999 Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Trans. Evol. Comput.* **3** (4), 257–271.

First received 25 October 2013; accepted in revised form 23 June 2014. Available online 15 July 2014