Chapter 10

Ensemble Prediction of Stream Flows Enhanced by Harmony Search Optimization

Milan Cisty and Veronika Soldanova

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.71192

Abstract

This work presents the application of a data-driven model for streamflow predictions, which can be one of the possibilities for the preventive protection of a population and its property. A new methodology was investigated in which ensemble modeling by data-driven models was applied and in which harmony search was used to optimize the ensemble structure. The diversity of the individual basic learners which form the ensemble is achieved through the application of different learning algorithms. In the proposed ensemble modeling of river flow predictions, powerful algorithms with good performances were used as ensemble constituents (gradient boosting machines, support vector machines, random forests, etc.). The proposed ensemble provides a better degree of precision in the prediction task, which was evaluated as a case study in comparison with the ensemble components, although they were powerful algorithms themselves. For this reason, the proposed methodology could be considered as a potential tool in flood predictions and prediction tasks in general.

Keywords: time series of river flows, ensemble prediction, optimisation, harmony search, data-driven methods

1. Introduction

Effective water resources management is one of the most crucial environmental challenges of our time. The inundation and flooding of landscapes and urban areas are serious problems, which cause immense damage to infrastructures and human lives in various parts of the world (e.g., recently in Australia, South America, Pakistan, West Africa and China, just to mention a few). Flood prevention requires various management tools, among which flow

INTECH
open science | open minds

prediction models occupy an important place. Flood warnings several days in advance could provide civil protection authorities and the public with the necessary preparation time and could reduce the socio-economic impacts of flooding [1].

This work presents the application of a data-driven model for streamflow predictions, which can be one of the possibilities for the preventive protection of a population and its property. There are various types of models for flow predictions: physically based, conceptual and data-driven models are among the most well known. While physically based models mainly depend on our knowledge of the physical laws in a watershed and on the corresponding geographical database, which serve as an information background for the application of the physical laws, data-driven models extract knowledge only from the monitored data describing the inputs and outputs of the watershed, e.g., time series of precipitation, temperatures, river flows, etc. For this reason, data-driven models are much more suitable for this task. It is not possible operatively to update all the detailed information about a watershed and its stated variables on a day-to-day or even hour-to-hour basis, which is necessary in the case of the application of physically based models.

The authors of this paper have focused on the application of a supervised learning methodology for flow prediction, namely, on a proposed ensemble approach, with the aim of refining the precision of the results of such modeling. In a typical supervised learning scheme, a set of input data instances, also referred to as a training set, is given. The output values of these data in the training set are known, and the goal is to construct a model in order to compute the outputs for the new instances (where the outputs are unknown).

Various models frequently show different capacities to maintain certain aspects of the hydrological processes [2], so the application of a single model often leads to predictions that could be more precise in some part of the problem domain but are less suitable in others [3].

The recognition of this fact has led to the application of an ensemble or committee of models being simultaneously considered. Many researchers have shown that by combining the output of many predictors, more accurate predictions can be produced than what could be obtained from any of the individual predictors [4–6]. Individual predictors should be accurate enough and also different from each other [7–9]. Sampling different training datasets, using different learning architectures and using different subsets of variables are the most popular approaches used to achieve such diversity [5, 10] in the application of the data-driven modeling approach. For example, in bagging [4], each classifier is trained using a different training set sampled from all the available training data. Boosting algorithms are different and powerful ensemble learners, which implement forward stagewise additive modeling, where in each stage the data are reweighted: the examples that produced the worst predictions gain weight and the examples that produced precise results lose weight. Thus, the next basic learner is focused more on examples that were previously incorrectly predicted. Stacking, another type of ensemble learner concept, tries to learn which base models are more reliable than others by using a meta data-driven algorithm, the task of which is to discover how to best combine the output of the base models to achieve the final results.

In the field of streamflow forecasting, various papers have been published [3] in which the data-driven ensemble modeling approach has been studied, but they are usually focused on climate inputs obtained by ensemble modeling of weather, which is not the subject of this paper. Selection of existing works from the focus of this article follows.

The application of a modular approach that uses different neural network rainfall-runoff models according to the hydrologic situation in a catchment was presented in Ref. [11]. A specific model from a set of trained models is proposed here to apply to particular input data. This work proposes that the model used for particular inputs is chosen on the basis of the most similar hydrological and meteorological conditions used to train the selected model. A clustering technique based on self-organizing maps was applied to manage the model's selection. A boosting application is presented in Ref. [12], where the authors demonstrated the advantages of an improved version of boosting, namely, AdaBoost.RT, which is compared to other learning methods for several benchmarking problems, and two problems involving river flow forecasting. In a recent study [13], the authors investigated the potential usage of bagging and boosting in building classification and regression tree ensembles to refine the accuracy of streamflow predictions. They report that the bagged model performs slightly better than the boosted model in the testing phase. An ensemble neural network (ENN) designed to monthly inflows forecasting was applied in Ref. [14] to prediction of inflows into the Daecheong Dam in Korea. The ENN combined the outputs of the members of a neural network employing the bagging method. The overall results showed that the ENN outperformed a simple artificial neural network (ANN) among the three rainfall-runoff models. Cannon and Whitfield [15] studied the use of ensemble neural network modeling in streamflow forecasting. Boucher et al. [16] used bagged multi-layer perceptrons for the purpose of a 1-day ahead streamflow forecasting on three watersheds.

In general, the ensemble methods as described in the published theoretical and application papers are usually composed of weak predictors, e.g., decision trees or neural networks commonly used as base predictors while building ensemble machine learning models. On the other hand, there are only a few works in which the ensemble is formed by a fusion of strong learners. The authors of the present paper assume that it is also important to examine ensembles based on nonweak learners, such as support vector machines, random forests or various other types of strong models, which are in some cases eventually ensembles themselves (composed of weak learners, e.g., various types of boosting methods).

A major goal of the analysis in this study is to precisely evaluate ensembles composed of various strong machine learning algorithms in comparison with the results achieved by individual learners. The final prediction by the proposed ensemble is accomplished by weighted summation of the results of the individual learners. The specification of these weights is a particularly important step in ensemble model building and is proposed to be solved with the help of the harmony search optimization methodology [17]. The harmony search methodol ogy has been successfully applied to various optimization tasks and also in the area of hydrology and water resources management, e.g. [18, 19].

In Section 2, the methods of the particular machine learning algorithms involved in this study are briefly explained, together with the ensemble methodologies used. Then, the data acquisition and preparation is presented. In Section 3, the settings of the experimental computations are described and the results are evaluated. Finally, Section 4 summarizes the main achievements and conclusions of the work and proposes ideas for future work in this area.

2. Materials and methods

2.1. Description of case study and preparation of data

Ensemble modeling by data-driven methods was applied for the 2-day ahead prediction of flows on the Hron River in Slovakia. The watershed of this river is a sub-basin of the Danube River. This task was accomplished by using data observed in the period from 01-01-1984 to 31-12-2000. Specifically, the average daily flow [$m^3\ s^{-1}$], the average daily temperatures [°C] and the daily rainfall depths [mm] were used.

The prediction of flows at the Banska Bystrica gauging station (**Figure 1**) serves as the case study in this paper. Each row in the input file for this task includes the date of the predicted flow, the predicted flow itself (two days' ahead—these are the modeled data, but their values are necessary for the training and testing mechanism), the input data of the flows from the three measuring stations, the temperatures from five meteorological stations, and the precipitation from 51 stations. All the input data were included in the input dataset from 1, 2, 3 and 4 days before the date of the predicted flow. This means that a summary of 238 variables is in each data row. Because daily data were used from 01-01-1984 to 31-12-2000, 6209 rows are in the dataset.

Some data preprocessing procedures had to be accomplished: cleansing the data, formatting it, inputting the missing data and normalizing it. These operations are not described here, because they are common procedures in data mining. A few words will follow about the division of the data and the sampling, which were important from the point of view of this paper.

The correct prediction of high flows is the most important task for flood predictions. The period from 1996 to 2000 includes many situations with high flows and floods, which was the reason for its selection as the testing period. The rest of the years (1984–1996) were used for the training (**Table 1**).

A sampling of the data was also accomplished to obtain a balanced training dataset and dataset that led to less demanding requirements from the point of view of the hardware and

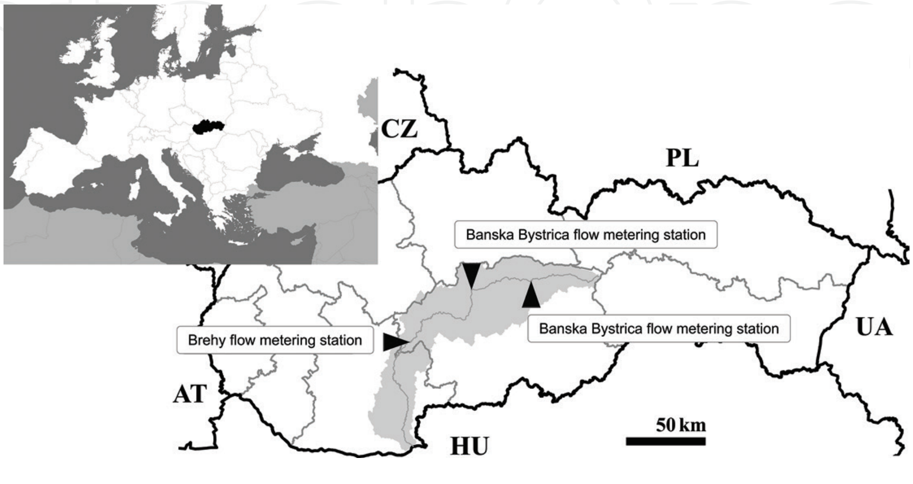
CZ
PL
UA
AT
HU
Banska Bystrica flow metering station
Banska Bystrica flow metering station
Brehy flow metering station
50 km

Figure 1. Map of the area studied within the Hron River watershed.

	Flows in Banská Bystrica [$m^3 s^{-1}$]			**Average temperatures—all 5 stations [°C]**			**Average precipitation—all 51 stations [mm]**		
Data	**All data**	**Training (84–96)**	**Testing (96–00)**	**All data**	**Training (84–96)**	**Testing (96–00)**	**All data**	**Training (84–96)**	**Testing (96–00)**
Min	5.18	5.18	5.29	−27.0	−27.0	−21.7	0.0	0.0	0.0
Max	219.20	219.20	157.90	27.6	27.3	27.6	123.6	123.6	93.5
Avg.	23.04	22.94	23.23	7.75	7.67	7.98	1.99	1.98	2.07

Table 1. Statistics of the data.

CPU time. Because of these computer power demands, sampling as a form of data reduction is a particularly important procedure in ensemble modeling, because in such modeling many runs of many algorithms are necessary, and computer demands rise with the amount of data used for training. A proper sampling methodology should be chosen in relation to the properties of the data and the problem studied. In streamflow predictions, a high amount of relatively low flows is usually available (also in the case studied), which led to the authors' decision to filter out some of them. On the contrary, high flows are somewhat rare. Because high flows are the most important data in flood predictions, the decision was made to filter out the data nonuniformly and leave all the input rows with this rare and large flow data in the final training dataset. Exactly, the same sampling of the same data was described in the previous work of the authors of the present paper [20], in which more details can be found.

2.2. Methodology

The goal of the proposed ensemble methodology is to combine the predictions of several models in order to improve the robustness/generalizability that could be obtained from any of the constituent models. The proposed ensemble methodology for predicting the river flows is divided into four equally important steps (**Figure 2**). The preparation of the data was described in a previous part of this chapter. This section follows two subsections: in the first, members of the ensemble are described, whereas the second subsection contains a description of each model's weight optimization by the harmony search methodology. The final model is predicted using the weighted average of the base learners in which these weights are used.

2.2.1. Selection and training of ensemble members

In contrast to the usual approach when ensemble consists of less powerful algorithms, the authors' intention was to evaluate the use of strong algorithms for members of the ensemble. The choice of "strong" algorithms is based on some papers, which evaluate existing data mining algorithms [21, 22].

A grid search combined with a repeated cross-validation methodology was used for finding the parameters of all the models included in the ensemble [6, 7]. In this approach, a set of each model's parameters from a predetermined grid is sent to the parameter-evaluating algorithm. A 5-times repeated 10-fold cross validation was used to find best parameters for the final

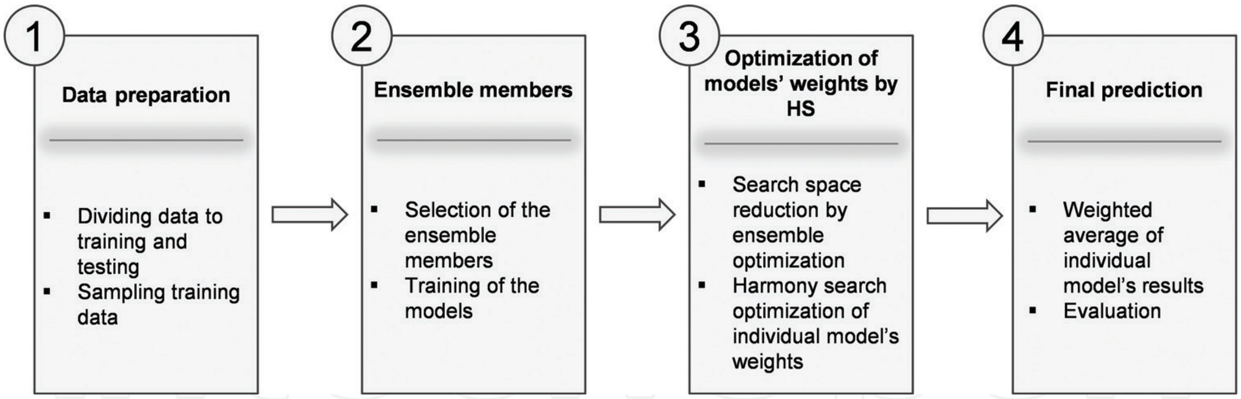
1
Data preparation
▪ Dividing data to training and testing
▪ Sampling training data
2
Ensemble members
▪ Selection of the ensemble members
▪ Training of the models
3
Optimization of models' weights by HS
▪ Search space reduction by ensemble optimization
▪ Harmony search optimization of individual model's weights
4
Final prediction
▪ Weighted average of individual model's results
▪ Evaluation

Figure 2. Proposed steps for the development of ensemble predictions of river flows.

models. Sampling of the data (as mentioned in the data preparation part of this chapter) was used in this process, because each basic algorithm runs in such a strategy many times.

Only a sketch of the algorithms is provided hereinafter, because in this work, a number of algorithms are used, and it is neither possible nor useful in this paper to go into a more thorough explanation. In the case of interest, the authors have indicated links to the relevant literature for detailed information.

2.2.1.1. Support vector machines (SVM)

A support vector machine (SVM) [23] is very effective, supervised, machine learning method for various machine learning tasks. It is specific by using kernel trick-nonlinear mapping used to transform the original training data of a nonlinear problem (which is also our case) into a higher dimension. Herein, SVM learn a nonlinear function indirectly and easier: they learn a linear function in the space induced by the particular kernel, which matches to a nonlinear function in the original space.

The next important concept in SVM methodology is to fully ignore small errors. In SVM, bounds for regression are set by defining the loss function that ignores errors, which are situated within the distance ε of the true value. This type of function is called epsilon insensitive loss function. As a consequence, good generalization of SVM is gained, because not all the input vectors of data are used, but only the so-called support vectors, which are training samples that lie outside of the boundary of the ε-tube.

In this chapter, the ε-SVM model was created by: (1) choosing a radial basis kernel with parameter sigma = 0.0005; (2) specifying the ε parameter to be equal to 0.1 and (3) specifying the capacity $C = 10.5$. All parameters were found by a grid search.

Multilayer perceptron (MLP).

Artificial neural networks (ANNs) are the most popular and well-known data-driven methodology; it has been described and is available in various literature sources, e.g. [24]. Briefly summarized, a multilayer perceptron, the most commonly used type of neural network, which was used also in this work, consists of input, hidden and output layers, all of which contain some processing elements or neurons. Input and output layer contains as many neurons as the model has input, respectively output variables. The so-called learning involves determination of number, types and particular properties of neurons in hidden layer. This layer is used for the transformation of the inputs to the outputs. A type of ANN known as a multi-layer perceptron (MLP), which uses a back propagation training algorithm, was used for generating the flow predictions in this study. The number of neurons in a hidden layer was found by a grid search and is equal to 6. Neurons with a logistic activation function were used in the hidden layer and with the linear activation function in the output layer.

2.2.1.2. Random forest (RF)

Random forests (RF) [25] are formed by a set of trees, which can either be classification or regression trees, depending on the problem being addressed. An RF prediction is an average of many trees (weak learners) grown on a bootstrap sample of the training data. The user chooses the number of trees in the forest (ensemble). Each tree is trained using a different bootstrap sample, which causes that different trees are obtained. For the regression task, the values predicted by each tree are averaged to obtain the final random forest prediction. In this work, a number of variables randomly sampled as candidates at each tree split were optimized with the help of a grid search, with the final value equal to 123. The minimum size of the terminal nodes is set at 5 and the number of trees at 500.

2.2.1.3. Multiple linear regression (MLR)

Multiple linear regression (MLR) analysis is generally used to find the relevant coefficients (*a*, *b*, *c*,…, *intercept*) in the following model:

$$Y = aX_1 + bX_2 + cX_3 + \ldots + intercept \tag{1}$$

This is a simple, well-known methodology, which the authors included in this paper mainly for the purposes of comparison with other, more powerful, methods.

2.2.1.4. Generalized linear model with an elastic-net (GLMNET)

Also in this method, as in previous case, a linear model is applied for flows prediction. Additional improvement in comparison to the basic multiple linear model is usage of regularization technique while searching parameters *a*, *b*, *c*,… from Eq. (1).

Regularization introduces additional criterion (or penalty) to the objective functions of optimization problems in order to prevent overfitting and for obtaining a more general model. In this case, least squares method for linear regression is meant as optimization problem. Various types of regularization exist. Ridge regression uses penalty, which limits the size of the coefficients in Eq. (1). Lasso uses a type of penalty which is trying to set some coefficients to be equal to zero. Elastic-net is a compromise between these two techniques and is used in this work. In work presented in this paper software provided by the authors of this regularization method was used [26].

2.2.1.5. Multivariate adaptive regression splines (MARS)

MARS [27] construct regression relations from a set of coefficients β and linear basis functions h that are determined from the training data. The general MARS model equation is given as:

$$y = f(X) = \beta_o + \sum_{m=1}^{M} \beta_m h_m(X) \tag{2}$$

The basis function $h(x)$ takes one of the following three forms:

1. A constant (the intercept).
2. A function of the form $max(0, x - const)$ or $max(0, const - x)$. MARS selects the values of *const* for the knots of this function. These breakpoints define the region of application for a particular linear equation.
3. A product of two or more of the above-mentioned functions. The model interactions between two or more variables are modeled in this case.

The best parameters of multivariate adaptive regression splines were found by a grid search procedure; the maximum degree of interaction is equal to 1, and the maximum number of terms (including the intercept) in the pruned model was found as 31.

In recent years, boosting has developed into one of the most important techniques for fitting regression models in high-dimensional data settings. So, the authors decided to include the proposed ensemble in the three boosting models described below. Boosting, or additive models [28], express the searched function as a weighted sum of the basis functions as follows:

$$f(x) = \sum_m \beta_m fm(x) = \sum_m \beta_m b(x; \gamma_m) \tag{3}$$

The basis functions b are dependent on the type of boosting method, and the parameters (β_m and γ_m) are assessed by minimizing a loss function (e.g. a mean square error) over the training data. Forward stagewise fitting is used for estimating β_m and γ_m sequentially from $m = 1$ to n. For example, for boosted trees with a squared error loss, we fit a least-squares regression tree to the residuals of the previous iteration.

2.2.1.6. Boosted linear models (B_GLM)

In this case, a linear model is fitted using gradient boosting, where the component-wise linear models are utilized as base learners. The methodology is described in Ref. [29]. In this work, the R package mboost and glmboost function with a default setting were used for this methodology [30, 31]. The number of initial boosting iterations was found by grid search and is equal to 150; shrinkage parameter was set to 0.1.

2.2.1.7. Gradient Boosting with Smooth Components (B_GAM)

A (generalized) additive model is fitted in this case using a boosting algorithm based on component-wise univariate base learners (where only one variable is updated in each iteration of the algorithm) in combination with the L_2 loss function. A spline, which is a sufficiently smooth polynomial function that is piecewise-defined, is suitable for this task. It possesses a high degree of smoothness at the places where the polynomial pieces connect (which are known as "knots"). In this study, P-splines with a B-spline basis [32] were used as a base learner. In each iteration of the gradient-boosting algorithm, a base learner is fitted to the negative gradient of the L_2 loss function. The current estimate of the predictor function is then updated with the actual estimate of the negative gradient, which automatically results in an additive model fit. In this work, the gamboost function of the R mboost package [33] was used to fit the flow prediction model. The number of initial boosting iterations was found by grid search and is equal to 100; shrinkage parameter was set to 0.1.

2.2.1.8. Gradient boosting machines (GBM)

Gradient boosting machines (GBM) are one of the most powerful boosting methods. Similarly to the other boosting methods, gradient boosting combines weak learners into a single strong learner. In GBM, decision trees (regression trees in our case) are usually employed. Weak learners are sequentially used with continually modified selection of the data. Moreover, training set is in this stepwise procedure weighted for current iteration according to the accuracy of the previously fitted model. The final prediction is obtained as a weighted average. Gradient boosting used in this work is implemented in the R package gbm [34] and is freely available. The total number of trees to fit is equal to 700 in this work and this parameter was found by a grid search. The shrinkage in GBM is controlled by parameter υ, which was set in this work to 0.01 (default value). Also, the maximum depth of the variable interactions was found by a grid search with up to 10-way interactions.

2.2.2. Harmony search (HS)

The harmony search [17] algorithm (HS) was adopted from the musical process of finding "pleasant harmonies" through improvisation. The five fundamental steps of HS could be summarized as follows:

> Step 1. Design the variables and initialization of the algorithm parameters. Initialization of HS search parameters: harmony memory size (HMS), harmony memory consideration rate (HMCR), the pitch adjustment rate (PAR) and the maximum number of improvisations (NI). The definition of the objective function $f(x)$, which has to be minimized (or maximized), is also performed in this step.
>
> Step 2. Initialization of harmony memory. The harmony memory is a memory location (matrix), where the solution vectors (sets of weights) and corresponding objective function values are stored. The initial HS memory consists of different randomly generated solution vectors.
>
> Step 3. The generation of a new harmony inspired by improvisation process in music is performed and accomplished in this step. New harmony represents new solution of given optimisation problem. It consists of three basic procedures: (1) selection of harmony from the memory controlled by parameter HMCR, (2) pitch adjustment (parameter PAR) and (3) a pick a random value with probability 1-HMCR. A more detailed description of these HS operators can be found in existing HS literature, e.g. [18].

Step 4. A new solution's objective function computation. If the new harmony has better value of the objective function than any harmony in the harmony memory, the worst harmony vector in harmony memory is replaced by this new harmony vector.

Step 5. Repeat from Step 3 to Step 5 until termination criterion is satisfied. In this work, the harmony search stops if there is no improvement in an objective function during the last 500 iterations or if the total (predefined) number of iterations is reached.

3. Results and discussion

In this section, the computation procedures, which are necessary for obtaining the ensemble model, are described. The ensemble model is proposed to have the following structure:

$$P^j_{ensemble} = \sum_{n}^{i=1} \beta_i * P^j_i \tag{4}$$

where β_i are the weights of the individual learners and P^t_i is a vector of predicted flows by model i for day j. The harmony search method was used to determine the corresponding weights of individual models. Application of this method for 2-day ahead prediction of flows follows in the subsequent paragraphs.

One harmony consists of n members, where n is the number of models. In the case of this work, there are nine models present in the ensemble. All values of the weights β_i are restricted to the interval $\langle 0, 1 \rangle$.

The problem solved should be defined by the objective function, which is proposed in this paper to have the following form:

$$O_f = 1 - \left(1 - \frac{\sum_{i=1}^{N} (O_i - P_i)^2}{\sum_{i=1}^{N} (O_i - \bar{O})^2}\right) + \left|\alpha - \sum_{i=1}^{n} \beta_i\right| \tag{5}$$

$$0 \le \beta_i \le 1, \tag{6}$$

where P_i and O_i are computed and observed flows, N is the number of days and $\bar{O}$ is average value of observed flows. Expression in the rounded parentheses is the Nash-Sutcliffe model efficiency coefficient (NSE). It was used in this study for evaluation of models efficiency because it is most often used to assess the predictive power of hydrological models. The NSE ranges from −∞ up to 1, where NSE = 1 means a perfect agreement between the observed and simulated data, i.e. closer the model efficiency is to 1, the more accurate the model is. The last component of the objective function (as an absolute value) forces the sum of the ensemble members' weights β_i to be equal to α, which is a regularization constant, by default equal to 1. Only rarely in cases when the models are systematically underestimating or overestimating, the regulation constant could have a slightly different values (maximum ±0.05). In this work, the authors only used the default value 1, because a relatively good prediction could be expected from the state-of-the-art models used as the ensemble members. This objective function is proposed to be minimized. In the case of an ideal model, the value of the objective function is zero.

Harmony search algorithm parameters were set as follow: *HMS* (memory size) was set to 10; *HMCR* (the harmony memory's consideration rate) was set to 0.91 and *PAR*, i.e., the pitch adjustment rate, was set to 0.1. The maximum number of improvisations $NI = 500{,}000$.

One of the main issues which must be carefully considered is what exactly has to be data P_i, which will serve as inputs to the harmony search optimization objective function (5). As was previously stated, these are basically the computed values of the predicted flows by each model. While performing these computations, we are in a model building phase, and that is why only training data can be used. There are two possibilities evaluated in this study as to how to obtain such data. The first possibility is achieved using the following steps:

1. The training data and repeated cross validation are used for finding the proper parameters of each model.
2. Every model (ensemble member) is trained with the values, which were found in step 1 with all training data.
3. The values of the predicted flows are computed by the models from step 2 from all the training data for each ensemble member. The number of rows of resulted input matrix for HS $P_{R,C}$ is equal to the number of the rows of training data (535 in this study) and the number of columns $C = n + 1$ (n is the number of models, and one extra column is the observed data). In this work, $n = 9$.

The problem of obtaining data $P_{R,C}$ by this methodology, if it is used for calculating ensemble weights, is that in this approach there is no mechanism that avoids overfitting of the final ensemble. Overfitting or a lack of generalization means that the weights of the models obtained could work well on the training data, but poorly on the testing set. Due to this problem, the authors also proposed a second option, which will be compared to the previous one:

1. The training data and cross validation are used for finding the proper parameters of each model.
2. When these parameters are obtained, the $k - 1$ folds (in the case of a k-fold cross validation) are used for training with the best parameters, and 1 fold is computed by the model obtained as a test.
3. This is repeated k times for every model included in the ensemble.
4. Because the r-repeated cross-validation was proposed in this work, steps 2 and 3 are repeated r times.
5. The computed values from all such testing folds from the cross-validation are used as the input matrix for the optimization by HS, which is proposed for searching the weights of each model in the final ensemble.
6. Consequently, the inputs to the HS are de facto testing data, although from the training set (the results from the testing folds in the cross-validation). When n is the number of models in the ensemble, N is the number of data in the training set and r is the number of repeats of

the cross-validation, the number of rows of this input matrix $P_{R,C}$ is $R = N^*r$ and the number of columns $C = n + 1$ (one column is the observed data). In this work, $n = 9$, $k = 10$, $N = 535$ (the data were reduced by the sampling!) and $r = 5$.

The ensemble models obtained from these two approaches are hereinafter identified as EHS1 for the first case and EHS2 for the second.

The process of assessing the performance of a hydrologic model involves making some estimates of the "closeness" of the simulated behavior of the model to observations (in our case, the streamflow). The most basic approach for assessing a model's performance is through a visual inspection of the simulated and observed hydrographs (**Figure 4**). An objective assessment requires the use of a mathematical estimate of the error between the simulated and observed hydrological variables. The predictive accuracy of the ensemble and its members was evaluated using the Nash-Sutcliffe coefficient of efficiency (NSE), the root mean square error (RMSE) and the correlation coefficient (*r*).

In **Table 2**, the root mean square error, correlation coefficient and Nash-Sutcliffe efficiency are evaluated for the ensemble members and the proposed ensembles. The identification of the models from their abbreviations in the heading of this table is possible. Two ensemble optimization approaches, which are identified as EHS1 and EHS2, are evaluated in **Table 2** and were described hereinbefore.

The selection of the appropriate settings for the ensemble members evaluated in **Table 2** is described in Section 2.2. A grid search was mostly used for the tuning; in some cases, the settings recommended in the scientific literature were applied. Regarding ensembles EHS1 and EHS2, it can be clearly seen that the hypothesis about the poor performance of the above-mentioned first proposition for obtaining matrix $P_{R,C}$ was confirmed. Ensemble model EHS1 performed well on the training data (with an NSE equal to 0.82, when an NSE of 0.79 was achieved by the best ensemble component, which was the GBM model), but on the testing set, which is evaluated in **Table 2**, the ensemble EHS1 gives worst results than most of the ensemble members. The ensemble approach to modeling is worth applying only in a case where the ensemble performs better than any of its members. If one considers the weights of the multilayer perceptron in ensemble EHS1, it is presumably inappropriately high (MLP are generally less precise

	GBM	B_GLM	RF	MLP	MARS	MLR	SVM	B_GAM	GLMNET	EHS1	EHS2
NSE	0.806	0.783	0.808	0.676	0.593	0.376	0.800	0.787	0.782	0.759	0.825
r	0.898	0.885	0.900	0.832	0.802	0.724	0.896	0.888	0.884	0.874	0.909
RMSE	13.575	14.371	13.519	17.548	19.661	24.355	13.788	14.219	14.410	9.684	8.247
Weights EHS1	0.128	0.011	0.190	0.549	0.021	0.022	0.032	0.003	0.045		
Weights EHS2	0.134	0.056	0.379	0.034	0.083	0.021	0.218	0.029	0.046		

Table 2. Evaluation of the computations by ***r*** and NSE and the final values of the model weights in the ensembles.

models), which means that this model is overfitted and that the poor generalization is a consequence of the approach used for the development of the EHS1 model. To the contrary, according to **Table 2**, in which the testing data are evaluated, the results with a good generalization were achieved by ensemble EHS2. From now on, we will only speak about this second model.

Column nine of **Table 2** with the evaluation of the ensemble members could also be seen as a case study of the evaluation of these models. The models are ordered from best to worst, so they can be ranked and compared with each other. As could be expected for such a complicated process as the flow formation in a river is, this process was described more successfully by nonlinear models, especially by the recently developed boosting types of algorithms. However, when the weights of the models for the EHS2 ensemble in **Table 2** are considered, it can be seen that this order does not imply that the weights will also be ordered in the same way as precision. An efficient ensemble should consist of predictors that are not only sufficiently precise, but also diverse, i.e. ones that if make wrong predictions they make them at different parts of the input space, e.g. which are not highly correlated. The correlation of the models is evaluated in **Figure 3**.

From the conjoint consideration of **Table 2** (weights of models for the EHS2) and **Figure 3**, it can be seen that, after optimization of the weights, the best three models, the GBM, RF, and SVM, are included in the proposed ensemble with the highest contribution (their weights are the highest). But the next best model, the boosted GAM (B_GAM), is included in the ensemble with a relatively small weight. That is because this model is highly correlated with the three best models mentioned and also with the GLMNET model. A similar case could also be observed with some other members of the ensemble. From this phenomenon, it could be evaluated that the optimization procedure, which was proposed in this paper, is searching for the best weights not only from the point of view of the best performance of the models but also is considering the diversity of the models as well, which is, as was mentioned,

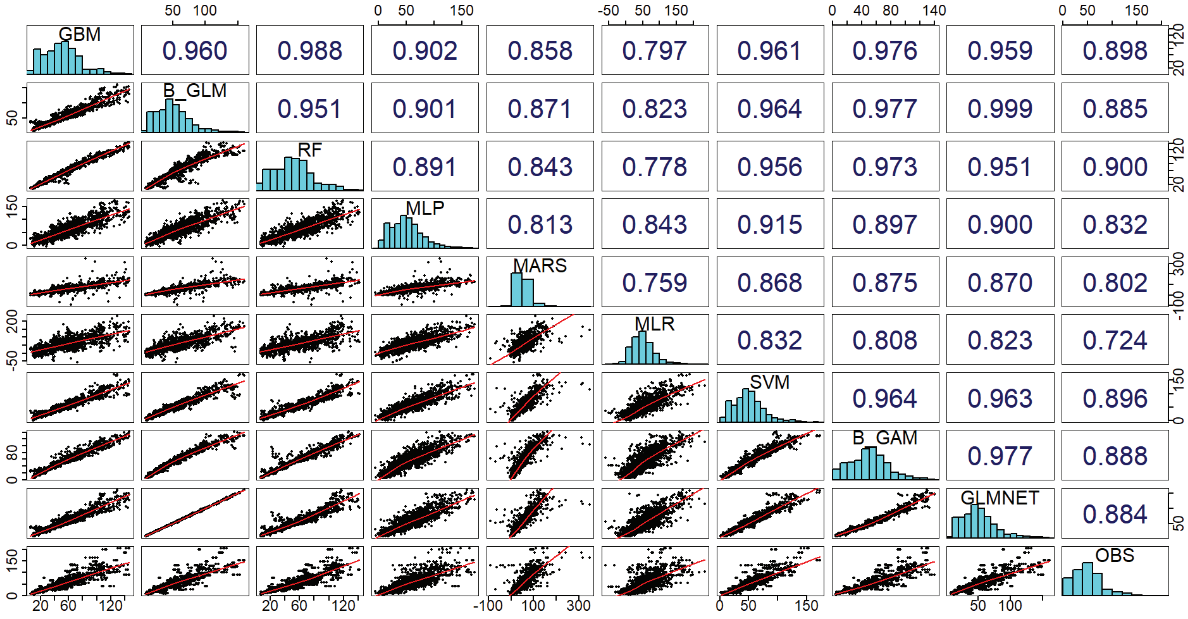

Figure 3. Correlation between the simulated results obtained by the ensemble members and with the observed data.

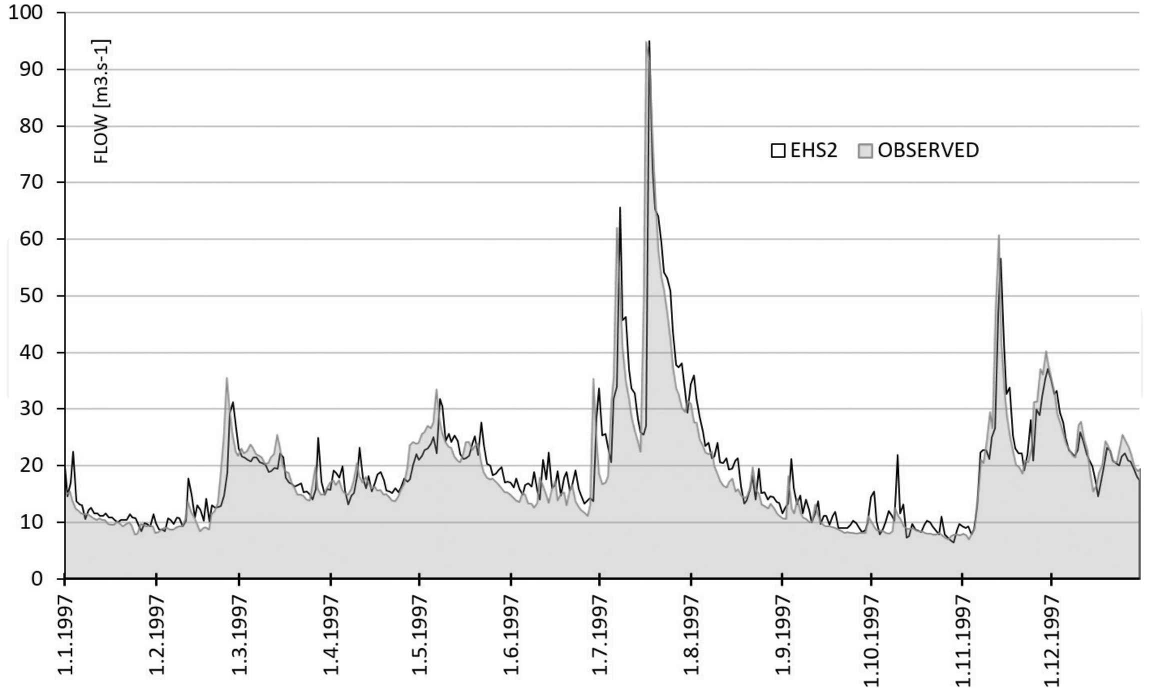

Figure 4. Time series of the testing dataset of the observed flows and the same flows simulated by the proposed ensemble model in the year 1997.

not less important. The authors assume that this is mainly due to the procedure by which matrix $P_{R,C}$ was obtained for model EHS2. As could be expected, the smallest contribution to the EHS2 ensemble has its least precise member: the multiple linear regression (MLR).

In **Figure 4**, a time series graph of the testing dataset with the observed flows and the flows simulated by the proposed ensemble model is seen. As can be seen, the predicted flows follow the real values with a high degree of precision, and the proposed ensemble approach could be used as an innovative alternative for flow predictions.

4. Conclusion

In this work, the authors deal with an investigation of the possible improvement of the river flow predictions. A new methodology was investigated in which ensemble modeling by data-driven models was applied and in which the harmony search was used to optimize the ensemble's structure. Because various data-driven models with strong prediction capability already exist, the authors were trying to evaluate in the case study presented in this paper (2-day ahead prediction of river flows), whether an ensemble paradigm would also bring some gain in cases when strong algorithms are used as ensemble members. Although the improvement in precision was not relatively as high as in the case when the ensemble consists of weak learners, it was proved that the ensemble model worked better than any of its constituents. These results mean, of course, that the proposed ensemble also works better than the ensembles with weak learners which are usually applied, because these were actually among the members of the proposed ensemble.

The authors' intention was to emphasize one important detail: how the input data for a harmony search optimization of weights should be properly computed. In the authors' investigation, it was verified that using the results of testing folds from cross-validation is the best option. This procedure is described in Section 3.

The authors like to emphasize the following practical aspect about ensemble modeling at the end of this paper. It is well known that for different datasets various algorithms may suit as best choice for prediction and it is never certain in advance, which one of these algorithms will perform with best results. This is known as "no free lunch" theorem. Because of this uncertainty, more algorithms must be usually trained, tested and evaluated during data mining process. These three activities (training, testing and evaluation) together with data preparation are quite laborious and computationally intensive. When this work is already done, instead of choosing only one of these algorithms for obtaining final results, it is wiser to use all already tuned algorithms for ensemble prediction of unknown variable (or subset of these algorithms). Updating prediction using ensemble paradigm almost always brings an improvement in precision as was also confirmed in the case study presented (the results are in **Table 2**). It does not mean a lot of extra work because tuned algorithms for a given task are already available. Gain will be different for different datasets, but as was confirmed also in this study it is surely worth to try this for such a little effort.

Acknowledgements

This work was supported by the Slovak Research and Development Agency under Contract No. APVV-15-0489 and by the Scientific Grant Agency of the Ministry of Education of the Slovak Republic and the Slovak Academy of Sciences, Grant No. 1/0665/15 and 1/0625/15.

Author details

Milan Cisty* and Veronika Soldanova

*Address all correspondence to: milan.cisty@stuba.sk

Department of Land and Water Resources Management, Faculty of Civil Engineering,
Slovak University of Technology in Bratislava, Bratislava, Slovak Republic

References

[1] Thielen J, Bartholmes J, Pappenberger F. Application of ensembles in flood forecasting. In: ECMWF Workshop on Ensemble Predictions; 7-9 November 2007; UK: Reading

[2] Cloke H, Pappenberger F. Ensemble flood forecasting: A review. Journal of Hydrology. 2009;**375**(3-4):613-626

[3] Duan Q, Ajami NK, Gao X, Sorooshian S. Multi-model ensemble hydrologic prediction using Bayesian model averaging. Advance in Water Resources. 2007;**30**(5):1371-1386

[4] Breiman L. Bagging predictors. Machine Learning. 1996;**24**(2):123-140

[5] Wheway V. Variance reduction trends on 'boosted' classifiers. Journal of Applied Mathematics and Decision Sciences. 2004;**8**(3):141-154

[6] Hastie TJ, Tibshirani RJ, Friedman JH, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer series in statistics. Springer: New York; 2009

[7] Krogh JV. Neural network ensembles, cross validation, and active learning. In: Tesauro G, Touretzky DS, Leen TK, editors. Advances in Neural Information Processing Systems; Cambridge, MA: MIT Press; 1995. p. 231-238

[8] Bacauskiene M, Verikas A. Selecting salient features for classification based on neural network committees. Pattern Recognition Letters. 2004;**25**(16):1879-1891

[9] Kuncheva LI, Whitaker CJ. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Machine Learning. 2003;**51**(2):181-207

[10] Bacauskiene M, Verikas A, Gelzinis A, Valincius D. A feature selection technique for generation of classification committees and its application to categorization of laryngeal images. Pattern Recognition. 2009;**42**(5):645-654

[11] Toth E. Classification of hydro-meteorological conditions and multiple artificial neural networks for streamflow forecasting. Hydrology and Earth System Sciences. 2009; **13**(9):1555-1566

[12] Shrestha DL, Solomatine DP. Experiments with AdaBoostRT, an improved boosting scheme for regression. Neural Computing. 2006;**18**(7):1678-1710

[13] Erdal IH, Karakurt O. Advancing monthly streamflow prediction accuracy of CART models using ensemble learning paradigms. Journal of Hydrology. 2013;**477**:119-128

[14] Jeong DI, Kim Y-O. Rainfall-runoff models using artificial neural networks for ensemble streamflow prediction. Hydrological Processes. 2005;**19**(19):3819-3835

[15] Cannon AJ, Whitfield PH. Downscaling recent streamflow conditions in British Columbia, Canada using ensemble neural network models. Journal of Hydrology. 2002; **259**(1):136-151

[16] Boucher MA, Laliberté JP, Anctil F. An experiment on the evolution of an ensemble of neural networks for streamflow forecasting. Hydrology and Earth Systems Science. 2010;**14**(3):603-612

[17] Geem ZW, Kim JH, Loganathan GV. A new heuristic optimization algorithm: Harmony search. Simulation. 2001;**76**(2):60-68

[18] Geem ZW, Tseng CL, Williams JC. Harmony search algorithms for water and environmental systems. In: Music-Inspired Harmony Search Algorithm. Berlin Heidelberg: Springer; 2009. p. 113-127

[19] Karahan H, Gurarslan G, Geem ZW. Parameter estimation of the nonlinear Muskingum flood routing model using a hybrid harmony search algorithm. Journal of Hydrologic Engineering. 2012;**18**(3):352-360

[20] Cisty M, Bezak J, Bajtek Z. Evaluation of the impact of the pre-processing of data on the effectiveness and accuracy of SVM. In: 13th International Multidisciplinary Scientific GeoConference SGEM. 2013;**2**

[21] Rich C, Niculescu-Mizil A. An empirical comparison of supervised learning algorithms. Proceedings of the 23rd International Conference on Machine Learning. ACM. 2006, Pittsburgh, USA. pp. 161-168

[22] Rich C, Karampatziakis N, Yessenalina A. An empirical evaluation of supervised learning in high dimensions. Proceedings of the 25th International Conference on Machine Learning. ACM. 2008; New York, USA. pp. 96-103

[23] Vapnik V. The nature of statistical learning theory, Springer-Verlag: New York; 1995

[24] Haykin SS. Neural Networks: A Comprehensive Foundation. Prentice Hall: Englewood Cliffs, NJ; 2007

[25] Breiman L. Random forests. Machine Learning. 2001;**45**(1):5-32

[26] Friedman J, Hastie T, Tibshirani R. Regularization paths for generalized linear models via coordinate descent. Journal of Statistical Software. 2010;**33**(1):1-22

[27] Friedman J. Multivariate adaptive regression splines (with discussion). Annals of Statistics. 1991:1-141

[28] De'ath G. Boosted trees for ecological modeling and prediction. Ecology. 2007;**88**(1): 243-251

[29] Buehlmann P. Boosting for high-dimensional linear models. The Annals of Statistics. 2006;**34**(2):559-583

[30] Hofner B, Mayr A, Robinzonov N, Schmid M. Model-based boosting in R: A hands-on tutorial using the R package mboost. Department of Statistics, Technical Report No. 120. 2012

[31] Buehlmann P, Hothorn T. Boosting algorithms: Regularization, prediction and model fitting. Statistical Science. 2007;**22**(4):477-505

[32] Schmid M, Hothorn T. Boosting additive models using component-wise P-splines as base-learners. Computational Statistics & Data Analysis. 2008;**53**(2):298-311

[33] Torsten H, Bühlmann P, Kneib T, Schmid M, Hofner B. Model-based boosting 2.0. The Journal of Machine Learning Research. 2010;**11**:2109-2113

[34] Ridgeway G. Generalized boosted models: A guide to the gbm package. Update 1.1. 2007