Flood Forecasting with Machine Learning in a scarce data layout

Theo Defontaine, Sophie Ricci, Corentin Lapeyre, E. Le Pape, Arthur Marchandise

PAPER • **OPEN ACCESS**

Flood forecasting with Machine Learning in a scarce data layout

Flood forecasting with Machine Learning in a scarce data layout

Théo Defontaine[1], Sophie Ricci[2], Corentin Lapeyre[2], Arthur Marchandise[3] and Etienne Le Pape[4]

[1] CERFACS, INP Toulouse, Toulouse, 31057 Cedex 1, France
[2] CERFACS-CECI, UMR 5318, Toulouse, 31057 Cedex 1, France
[3] Direction des Risques Naturels, DREAL Occitanie, Toulouse, 31074 Cedex 9, France
[4] Service Central d'Hydrométéorologie et d'Appui à la Prévision des Inondations, Ministère de la transition écologique, Toulouse, 31057 Cedex 1, France

Corresponding authors: defontaine@cerfacs.fr, ricci@cerfacs.fr, lapeyre@cerfacs.fr

Abstract. Flooding is one of the major natural disasters occurring in the world, with climate change increasing their occurrence and severity. Reliable flood forecasting models are needed to have better insurance in the emergency services' actions. This work reflects the capacities of Machine Learning models to improve discharge prediction results from empirical lag and route models based on hourly measured water level at gauge stations on the Garonne River. With scarce flooding data (30000 points) over the last 15 years, several learning algorithms have been implemented to predict floods in Toulouse at a 6-hour lead time from upstream stations providing hourly observations. A Linear Regression, a Gradient Boosting Regressor (Machine Learning) and a MultiLayer Perceptron (Neural Network, Deep Learning) are compared, using various strategies for learning, validating and predicting. Preliminary results show that the various strategies score as well as the empirical lag and route model. Further improvements are being investigated regarding the constitution of learning and validation data bases. This paper highlights how AI algorithms allow to improve the reliability of flood forecasts and how the layout of the limited volume of data influences their performance.

1. Introduction

As the effects of climate change become ever more significant, a surge of extremely large floods in European rivers has occurred in recent years. Accurate forecasts with long lead-times are crucial to give emergency services the opportunity to better react to these events and reduce their harmful effects. A good knowledge of various data on the geometry of the catchment, the properties of the land surface cover and the dynamics of the flow as in-situ observations of water level for instance is necessary to properly forecast flooding. The access to a large enough of good quality data remains a challenge.

This study was carried over the Garonne upstream of Toulouse, in south-western France. Full bathymetry and topography are not well known for this catchment as the river beds geometry strongly varies in the Pyrenees foothills and as bathymetry campaigns are still necessary. Fine hydrodynamics model solving Shallow Water Equations (SWE) such as MASCARET-TELEMAC [1, 2, 3, 4], is demanding and simpler approaches should be investigated. Rainfall-runoff model could be good candidate as rain data are available on the foothills but were not implemented so far for operational

purposes over this catchment [5]. Hydrology modeling in Toulouse is currently based on the use of upstream measurements at a large number of hydrometric stations on this catchment.

The local Flood Forecasting Service (*SPC*, Service de Prévision des Crues) in Toulouse has implemented an empirical model based on upstream timeseries in order to issue short-term forecasts at Toulouse station. This upstream part of the Garonne River is landlocked by surrounding mountains and the flow is linear enough to be predicted empirically at short lead times. Yet, the extension of the forecast lead time of such model is a key challenge as security and economics are at stake. In that perspective, the use of more advanced models based on learning from observed data time series is investigated here, for instance with Machine Learning (ML) approaches [6, 7]. It should be noted that the limited knowledge of the topography/bathymetry as well as scarce data layout for significant flood events over the catchments are constraints that heavily weigh on the learning strategy. The objective of this paper highlights how AI algorithms allow to improve the reliability of flood forecasts and how the layout of the limited volume of data influences their performance.

In the literature, several ML algorithms have been explored and preliminary conclusions have been drawn. For instance, the use of Recurrent Neural Networks (RNNs) as well as Long Short-Term Memory neural networks (LSTMs) are too demanding in terms of volume of data [8]. The use of Bayesian Linear algorithm [9] could be investigated. In the present work, classical ML algorithms were considered as preliminary study. Here, a Linear Regression (LR), Gradient Boosting Regressor (GBR) [10] and a MultiLayer Perceptron (MLP) are investigated in a scarce data layout. The performance and the robustness of the algorithms were assessed with multiple criteria, with a focus on the impact of the data layout, including outliers, within the learning and validation databases.

The remainder of this article is organized as follows. Section 2 described the catchment and its hydrodynamics, the data that are available for the study and how they could be used in quasi-operational mode. The ML models and their implementation are described in Section 3, along with assessment criteria. The results for water level and discharge estimation and forecast are detailed in Section 4. Conclusions and perspectives are finally given.

2. Study case

2.1. Study area and forecast model

The Garonne catchment in Toulouse shown in Figure 1, is equipped with a hydrometric station named *Toulouse Pont Neuf*, where observations are made hourly. This station represents a catchment of 10 133.95 km^2. There are several water height measurements (also hourly) stations upstream of Toulouse, each situated at a certain propagation distance and associated lead time from Toulouse. The stations of *Marquefave* (watershed area of 5 237.98 km^2 on the Garonne main branch) and *Auterive* (3 471.79 km^2 on the Ariege tributary) are close to Toulouse, and located on a reach that is characterized by a quasi)-linear dynamics of the flow. The propagation times to Toulouse are estimated by the Toulouse SPC to about 4h. The effect of the rain between these stations was shown to be negligible. For these reasons a lag and route model adapted to a 4h forecast was successfully calibrated at SPC Toulouse.

When longer lead time are considered, five stations should be considered upstream of Toulouse as shown in Figure 1: *Mancioux* (catchment area of 2 810.96 km^2, Garonne main branch), *Roquefort sur Garonne* (1 574.95 km^2, Salat tributary), *Mas d'Azil* (220.21 km^2, Arize tributary), *Saverdun* (1 813.83 km^2, Ariege tributary) and *Mazères* (1 376.84 km^2, Grand Hers tributary). The *SPC* has set an empirical lag and route flow forecasting model between those 5 stations and Toulouse. It has been calibrated specifically for 6h forecasts during floods. As the dynamics on this reach is less linear than that of reaches that are closer to Toulouse and as the number of tributaries is large, the calibrated model shows his reach shows fewer satisfying results. It should also be noted that the flow dynamics is now significantly influenced by rain fall, thus advocating for a more advanced modelling strategy for extended lead times and larger catchments.

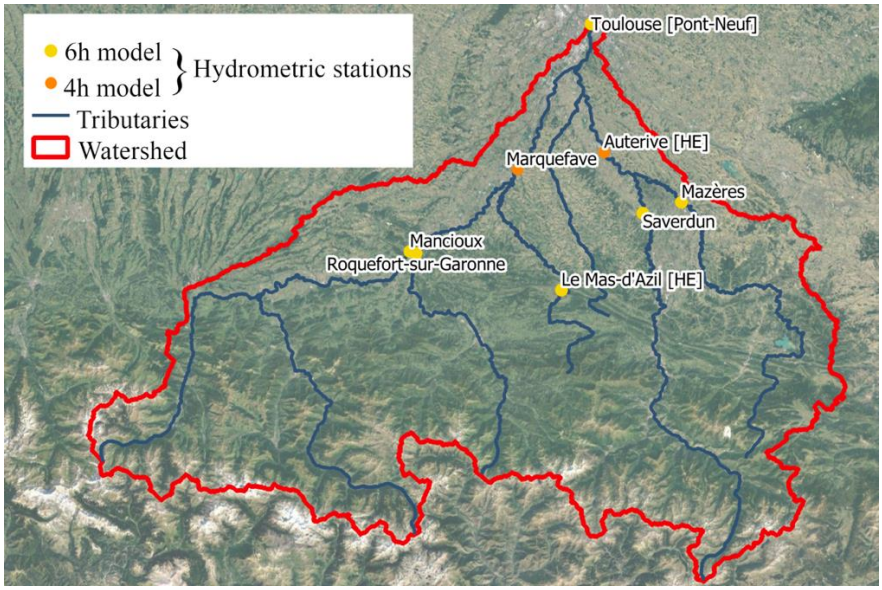

Figure 1. Garonne Catchment (in red) and its tributaries (in blue). Red dots are for the two hydrometric stations 4h upstream of Toulouse, yellow dots are for the five hydrometrics stations 6h upstream of Toulouse. Toulouse takes part in both models' outputs.

2.2. The database

The hydrometric stations used in this case provide hourly water level measurements throughout the year. Each of them has an experimental setting curve estimate by SPC from a limited number of discharge and height measurements on field, rarely in high flow. The rating curve is usually extrapolated for high flows, leading to a significant uncertainty in discharge estimates from water level measurement during flooding events.

All models that are considered in the present paper are calibrated for floods; the data sets are thus only composed of data measured during flooding events. Since 2005 (time at which the stations were consistently measuring), there has been about 40 flood events, half of them not rising the *SPC* vigilance level. The selection criterion for these events is based on a water level gauge in *Toulouse* which alerts the *SPC* when the threshold is exceeded. A preliminary study with 20 of those events from 2005 to 2015 showed promising results [11] for learning algorithms. Here, the database was extended so as to consider 30 flood events between 2007 and 2018. The data for these events were rearranged to fit those of a rainfall-runoff model calibration set with the Plathynes platform [12] at SPC for research purposes. These 30 flood events amount to 30000 data points available to train, validate and test the AI algorithms as each event lasts around 100 hours.

It should be noted that these events differ in various ways (severity, length, multiple/single peak, date, origin, ...) For instance, rain does not always fall on the same location of the catchment, with the same intensity, the input flow varies with the month of the year and consequently with different impact of the snow melt process from the Pyrenees. The database is thus heterogeneous and scarce, it is assumed here that it is representative of the possible flood event that may occur over the catchment (for less than centennial flooding).

2.3. The SPC model: a baseline

The model discussed here is the *SPC*'s **6h-model** (similar to the 4h model); it was calibrated at SPC with gauge measurements (including some minor events).

SPC has set an empirical lag and route model (similarly to [13]) based on the hypothesis that the upstream stations are all situated at a 6h propagation time from Toulouse. For each time step, they use

a 6h fixed flow lag (water levels are converted into discharge through local rating curves) for each upstream station before summing them as shown in equation (1):

$$\begin{aligned} \underset{t\to t+6}{\Delta} Q_{\mathrm{Tls}} &= Q_{Tls}(t+6) - Q_{Tls}(t) \\ &= a_{\mathrm{Mn}} \underset{t-6\to t}{\Delta} Q_{\mathrm{Mn}} + a_{\mathrm{Rq}} \underset{t-6\to t}{\Delta} Q_{\mathrm{Rq}} + a_{\mathrm{Ms}} \underset{t-6\to t}{\Delta} Q_{\mathrm{Ms}} + a_{\mathrm{Sv}} \underset{t-6\to t}{\Delta} Q_{\mathrm{Sv}} + a_{\mathrm{Mz}} \underset{t-6\to t}{\Delta} Q_{\mathrm{Mz}} \end{aligned} \quad (1)$$

with $a_{\mathrm{Mn}} = 0.91$, $a_{Rq} = 0.70$, $a_{Ms} = 3.00$, $a_{Sv} = 0.40$ and $a_{\mathrm{Mz}} = 0.65$.

SPC manually calibrated this model with 20 events in the 2005-2015 temporal window, selecting events for their features. Some of those events featured only one tributary contribution thus allowing the SPC to calibrate the corresponding tributary static coefficient. The five static coefficients shown in equation (1) are calibrated one by one with events that are mainly influenced by one. These coefficients compensate for the fact that rain input data are not accounted for.

2.4. Experimental settings

As seen in subsections 2.2 and 2.3, the *SPC* model is fed with flows in input, through setting curves. It also brings some uncertainties. To see the extent of their reliability, an experiment was made for the AI algorithms without the setting curves, named E_h in table 1. The experiment without the curves is named E_Q in the following table 1.

The learning experiments also differ from the data base setting as shown in Figure 2. In the following, for each learning algorithm, the data was split into a training set (~80% of the whole dataset) and a testing set (~20%). When the events in the learning data base are selected chronologically, meaning that whatever happens in the future is caused by what happened in the past, the experiment is called E_c (in table 1). In this case, the test set is composed of the 6 last events. When the events in the learning data base are taken arbitrarily, the experiment is called E_a. Figure 2 shows the repartition of the events in the sets. In this latter case, the validation set is similar to that in [12].

Table 1. Specific designation of the modeling experiments.

Experiment	E_Q	E_h
E_c	**dqc**	**dhc**
E_a	**dqa**	**dha**

The experiments here are related to two modifications of the setup: one being the choice of the testing set (E_c and E_a in table 1), the other being the choice of the kind of quantity learned (E_Q and E_h in table 1). Each experiment has thus two variations (one for each experiment related to the other modification).

Figure 2. Location of test set events for E_c and E_a. Orange coloured dots stands for training events, green for test events.

It should be noted that the input data are considered as instantaneous values. The treatment of the temporality of the data within the learning process is beyond the scope of this paper.

3. Machine Learning algorithms and validation criteria

3.1. Machine Learning Algorithms

Three different Machine Learning algorithms were implemented in python 3 using scikit-learn [14] and TensorFlow with Keras [15].

3.1.1. Linear Regression. For this scikit-learn's *linear_regression* module, basic settings were used. Here, the output y_i is supposed to be a linear combination of the inputs x_i plus random noise ϵ_i, as shows equation (2):

$$\forall i \in [\![1,n]\!], \qquad y_i = \sum_{i=1}^{n} x_i \alpha_i + \epsilon_i \quad with\ \epsilon_i \sim \mathcal{N}(0,\sigma^2) \tag{2}$$

where α_i and σ are à priori unknown. α_i being the weights to be determined. The σ are supposed to be independent of each other. This scikit-learn algorithm aims at minimizing the quadratic error norm of the weights. It is a least squares regression. As expected, this simple algorithm performs poorly when tackling complex processes.

3.1.2. Gradient Boosting Regressor (GBR). The idea behind the gradient boosting algorithm is to minimize the error of a chosen simple estimator on the observations. To do so, other simple estimators are used recursively to correct the error of the previous estimator. The recursion follows a steepest gradient descent. The whole mathematical process is described in [16].

With this more complex algorithm, ranges of operability are used for a subsample of the hyper-parameters that may be optimized as explained in Section 3.2. The scikit-learn version of this algorithm was used involving the Python class *GradientBoostingRegressor*. Two different setups are implemented, with *absolute error* metric and loss:

- The first setup takes as input the data from the upstream stations and tries to minimize the *absolute* error relative to the observations in Toulouse.
- The second setup takes as input the upstream observations together with the output of the *SPC* model and the *Linear Regression*. The main purpose of this setup is for the *GBR* to correct the outputs of the other two given as inputs. This setup makes the *GBR* act as an error correction algorithm. In the following, this algorithm will be referred to as *GBR+*.

3.1.3. MultiLayer Perceptron (MLP). This *Neural network* is the simplest multilayer Deep Learning algorithm. Its implementation is done with Keras [15, 17] (with the *TensorFlow* underlying system). It had 3 to 4 fully connected layers (1-2 hidden), with up to 100 nodes each. The last layer was a single node layer to have a scalar output.

The choice was made to convert the basic *Sequential* algorithm into an *estimator*. It is then easier to use other ML tools (especially those from scikit-learn). With this DL algorithm, ranges of operability are used for a subsample of the hyper-parameters that may optimized as explained in Section 3.2. Here, two different setups were also implemented, the same ones as explained for the *GBR*, with *absolute error* metric and loss*:*

- The first setup takes as input the data from the upstream stations and tries to minimize the *absolute* error relative to the observations in Toulouse.
- The second setup takes as input the upstream observations together with the output of the *SPC* model and the *Linear Regression*. The main purpose of this setup is for the *MLP* to correct the outputs of the other two given as inputs. This setup makes the *MLP* act as an error correction algorithm. In the following, this algorithm will be referred to as *MLP+*.

3.2. Optimization process

For GBR and MLP, an optimization process was implemented. *RandomizedSearchCV* from scikit-learn was investigated. It consists in giving each hyperparameter a range of values to be tested during training. To limit overfitting on the training set with this method, a cross validation step during training is mandatory.

3.2.1. Cross Validation. Cross validation is used during training to prevent optimized algorithms from overfitting. A grouped cross validation algorithm was used to prevent events being cut in parts. The *GroupKFold* (from scikit-learn) was splitting the training data into 5 groups that were used one by one as a validation set (one group) and training set (the remaining 4 groups). Thus, there is 5 different training plus validation splits on which we can train the algorithms.

3.2.2. Randomized Search. Following a Monte Carlo method, several algorithms are tested at once. The sampled algorithms have randomly chosen hyperparameters in the ranges given for them. With enough samples over the ranges given, it is considered that the best found hyperparameters are close to optimal ones that could have been found by searching exhaustively over the ranges [18]. This optimizer uses a Cross-Validation as explained above, it trains on one set of hyperparameters over the 5 different splits and then calculates the estimator's average metric value for this set of hyperparameters.

3.3. Performance Criteria

For the training/validation, as shown above, each algorithm had its own metric. For better comparison, the same 3 criteria were used to assess their performances on the test set and on the result of the training. In this section, every time series is considered discrete and thus numbered in the range $[\![1, n]\!]$. Each criterion was also computed in two different ways: by test/training set (a set is a list of several events) and by event.

3.3.1. Coefficient of determination (R^2 score). *This first criterion has the* SPC *model as a baseline and computes the relative squared error of the tested algorithm with respect to the reference model:*

$$R^2(\tilde{Q}, \hat{Q}) = 1 - \frac{\sum_{i=1}^{n}(\tilde{Q}_i - \hat{Q}_i)^2}{\sum_{i=1}^{n}(\tilde{Q}_i - \bar{\tilde{Q}}_i)^2} \tag{3}$$

where $\hat{Q}_i$ is the predicted value, $\tilde{Q}_i$ is the *SPC* baseline and the mean of the reference is $\bar{\tilde{Q}}_i = \frac{1}{n}\sum_{i=1}^{n}\tilde{Q}_i$.

Computing this over an event implies that n is the length of the event. When computing over a set of several events, *n* is the sum of the length of the events. When computed by set is also not equal to the mean of the corresponding events values. It variates between 1 (perfect prediction) and $-\infty$ (poorest prediction). A value of 0 means that the target has a mean value equal to the mean of the reference.

3.3.2. Nash-Sutcliffe Efficiency coefficient (NSE). This criterion formulates the relative error of an algorithm with respect to the observed flow. It reads:

$$NSE(Q, \hat{Q}) = R^2(Q, \hat{Q}) = 1 - \frac{\sum_{i=1}^{n}(Q_i - \hat{Q}_i)^2}{\sum_{i=1}^{n}(Q_i - \bar{Q}_i)^2} \tag{4}$$

where $\hat{Q}_i$ is the predicted value, Q_i is the *observed flow* and $\bar{Q}_i = \frac{1}{n}\sum_{i=1}^{n}Q_i$. The *NSE coefficient*'s equation is the exact same as the R^2 *score*, the only difference being the reference data, here it is the

observed flow, whereas with the R^2 *score* it is the *SPC model*'s results. This criterion has the same behaviour as the R^2 *score*.

3.3.3. Persistence criterion. This criterion computes the temporal shift between the modelled value and the observed value, relatively to a predetermined shift as shown in equation (5):

$$Persistence(Q, \hat{Q}) = 1 - \frac{\sum_{i=\Delta j}^{n}(\hat{Q}_i - Q_i)^2}{\sum_{i=\Delta j}^{n}(Q_{i-\Delta j} - Q_i)^2} \quad (5)$$

where $\hat{Q}_i$ is the predicted value, Q_i is the *observed flow* and Δj the predetermined shift of the time series, which is 6 in our case. For this criterion, the behaviour in terms of values obtained is the same as the previous criteria. In terms of computation, the criterion is only obtained per event. It would be meaningless to compute it over a set as it is time-dependent. The set score is thus the mean of the events scores.

4. Results and discussion

4.1. Machine Learning Algorithms' performances in the different experiments

Figure 3 displays the criteria values (test and training) for all 4 experiments' configurations. The metrics on the left (a), (c) and (e) are computed by set (training in light orange/test in dark green) for each one of the 4 experiments' configurations given in table 1. The metrics on the right (b), (d) and (f) are computed by event for the same experiments. Light orange dots indicate events that were used during training and dark green dots indicate events that were used for test. R^2 scores are plotted between -1 and 1. The other metrics are plotted between 0 and 1. It was observed that all three ML algorithms behave in the same way for the different experiments. For clarity purposes, only results for GRB+ are shown in the following, with loss of generality in the conclusions.

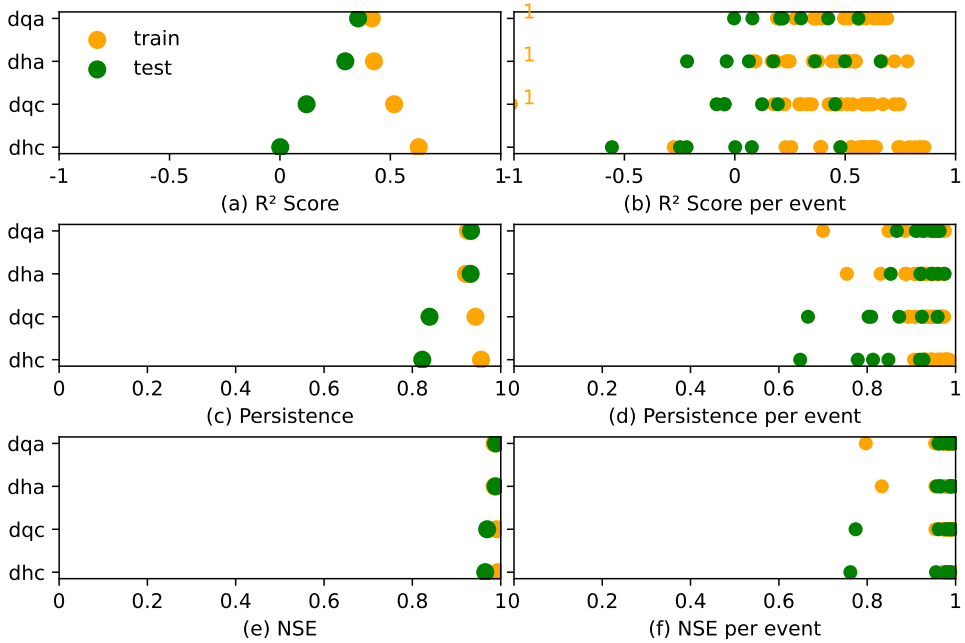

Figure 3. *GBR+ Algorithm*. Comparison of the different experiments' performances with respect to the various criteria. In (a), (c) and (e) each experiment is represented on a line (dqa, dha, dqc and dhc), test score in green, training score in orange. The orange line shows that training scores better than test. In (b), (d) and (f) the same scores are computed by event and grouped in colours by test/training set.

Figure 3 shows slightly better results for E_a with even better results in test than in training. This can be explained by the fact that more recent events (since June 2018) are less similar to the older ones, thus testing on them while training on older ones deteriorates measured performances (chronologic configuration). An outlier also seems to be present in the causal experiment E_c test set, visible on figure 2 (d) and (f). Arbitrarily chosen test events also appear to be among the most favorable ones in the dataset when it comes to scoring. They seem to be amongst the typical events on which the algorithms are well trained because they are well represented in the database.

The second part of these results show that there are higher uncertainties when learning directly from heights (dha, dhc). It confirms the *SPC*'s opinion that their rating curves are calibrated well enough and do not lead to significant errors. Plus, learning with flow variables is safer as it is derived from the mass balance equation. In this case, that equation is not closed because rain inputs are not taken into account and only balanced in the coefficients of the algorithms, thus uncertainties still remain.

4.2. Comparing algorithms performances for the same experiment

The most favourable strategy *dqa* is further investigated here, considering all available algorithms. Table 2 shows that the *GBR+* is the best scoring version of our algorithms, and the most consistent in that configuration. This result is not surprising as Gradient Boosting algorithms are easier to calibrate and have better manoeuvrability than MLPs in a scarce data layout.

The algorithms are in a favourable configuration as most test scores are above training scores in table 2. This is unusual for Machine learning algorithms. This can be explained by the low outlier that shows up in the training set. It is visible in figure 3 (b) where the orange number 1 shows that one event has performances worse than -1. The persistence and NSE scores of *dqa* in figure 3 (d) and (f) also shows a training event lower than others.

Another important feature shown in table 2 is that our algorithms all seem to outperform the *SPC model*. This was a targeted feature, our algorithms being more complex, it is not that surprising to see them perform better. It also shows the robustness of their implementation as they outperform SPC in all 3 criteria.

Table 2. Performances of the various algorithms in the *Arbitrary, flow experiment* (**dqa**)

Model	R^2 Score		Persistence		NSE	
	Train	Test	Train	Test	Train	Test
SPC	0	0	0.867	0.902	0.976	0.983
LR	0.237	0.325	0.898	0.929	0.982	0.989
GBR	0.344	0.274	0.916	0.926	0.984	0.988
GBR+	0.417	0.354	0.926	0.932	0.986	0.989
MLP	0.255	0.279	0.904	0.927	0.982	0.988
MLP+	0.251	0.272	0.904	0.926	0.982	0.988

4.3. Scarce data effects

Figure 4 gives a focus on a chosen test event with the configuration *dqc* (see table 1). It appears that all model fails to properly forecast this event (observations are plotted in black), similarly to the SPC model forecast represented in red. As the inputs upstream of Toulouse show limited flow rise, the flood peak is either due to some upstream influence that is not accounted for in the list of selected stations or to influence of rainfall in-between observing stations. This type of event and its influence in the learning and test sets should be further investigated. Indeed, the models outperforms with significant errors: this event has a mean absolute error (MAE) of 20cm and a maximum error of 109cm for the SPC model (the worst one) and a MAE of 19.69 cm, maximum error of 96cm for the *extended GBR*. These are way above average (less than 10 cm for MAE, less than 50cm for maximum error).

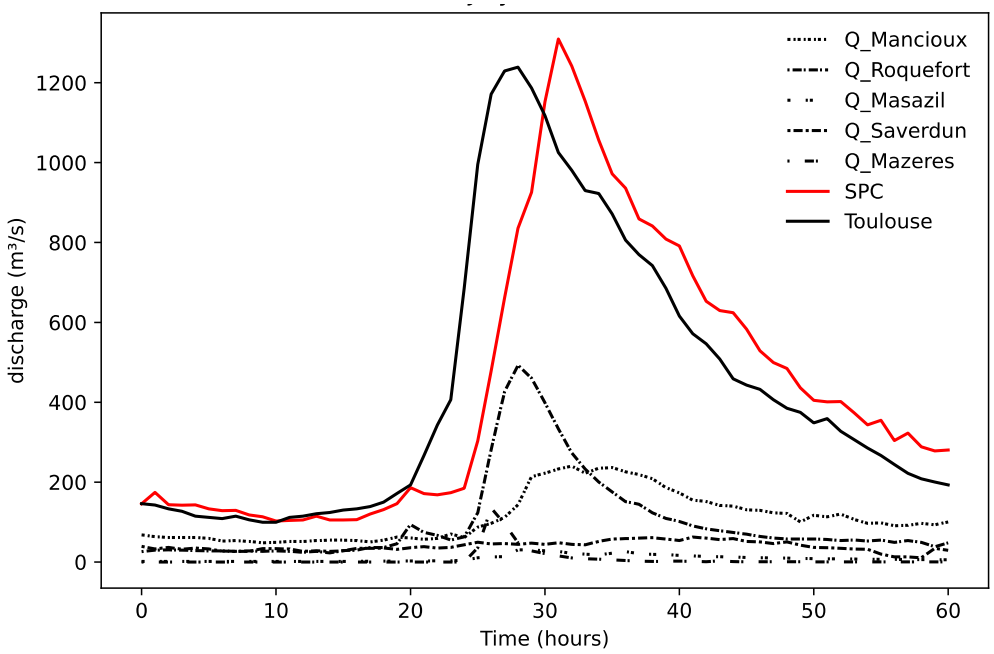

Figure 4. 15 July 2018 event. Observed flow inputs with a 6h-lag and flow output are shown (with the *SPC model*'s result for comparison).

5. Conclusion

Even though the data is scarce in this problem, good performances can be achieved with these models, with even better results than the operational model of the *SPC*. Attention must be put on the robustness of the algorithms used when the dataset is scarce as it is the case here. The experiments show that working with variables derived from a closed equation has better results, as the information then gathered by the algorithms is more robust. The scarcity also makes the problem less stable. The experiments conducted here also shows that there is a real influence of the choices made to separate the data into sets. Thus, this must be done with the most specific attention. There still lies some outliers that require to be better treated. They must be identified and the results treated with this knowledge as their influence is non negligeable. It can be wise to gather non global results to tackle these two points.

Although the dataset is pretty restricted, convincing results with simple algorithms are obtained. The scarcity of the events is the major problem here, making the algorithms less stable. Indeed, when complexity is added, most of the algorithms are bond to become divergent. Thus, any complexity add-on in a scarce layout must be done carefully. Complexity and scarce data do not fit well together.

To improve results, the authors aim to input more information, such as rain data or temporal window for data use. In both cases, the learning task becomes more complex and the stability and robustness of the algorithms becomes more challenging. The preliminary results obtained here are promising, with room for improvement.

References

[1] Tiberi-Wadier A L Goutal N Ricci S Sergent P Monteil C 2019 Sensitivity Analysis of the Mascaret Model on the Odet River *XXVIth Telemac & Mascaret User Club*

[2] Broich K Pflugbeil T Disse M Nguyen H 2019 Using Telemac-2D for Hydrodynamic Modeling Rainfall-runoff *XXVIth Telemac & Mascaret User Club*

[3] Barthelemy S Ricci S Rochoux M C Le Pape E Thual O 2017 Ensemble-based data assimilation for operational flood forecasting – On the merits of state estimation for 1D hydrodynamic forecasting through the example of the "Adour Maritime" river *Journal of Hydrology 552* pp 210-224

[4] Barthelemy S Ricci S Morel T Goutal N Le Pape E Zaoui F 2018 On operational flood forecasting system involving 1D/2D coupled hydraulic model and data assimilation *Journal of Hydrology* pp 623-634
[5] Todini E 1988 Rainfall-runoff modeling – Past, present and Future *Journal of Hydrology* pp 341-352
[6] Dawson C D Wilby R 1998 An artificial Neural Network approach to rainfall-runoff modeling *Hydrological Sciences Journal* pp 47-66
[7] Toukourou M Johannet A Dreyfus G Ayral P-A 2011 Rainfall-runoff modeling of flash floods in the absence of rainfall forecasts: the case of "Cévenol flash floods" *Applied Intelligence, Springer Verlag (Germany)* pp 178-189.
[8] Kratzert F Klotz D Shalev G Klambauer G Hochreiter S Nearing G 2018 Rainfall-runoff modeling using Long Short-Term Memory (LSTM) networks *Hydrology and Earth System Sciences* pp 6005-6022.
[9] Noymanee J Theeramunkong T 2019 Flood Forecasting with Machine Learning Technique on Hydrological Modeling *Procedia Computer Science* pp 377-386
[10] Sanders W Dongfeng L Wenzhao L Zheng N F 2022 Data-driven Flood Alert System (FAS) using Extreme Gradient Boosting (XGBoost) to forecast flood stages *Water 14 number 5: 747* https://doi.org/10.3390/w14050747
[11] Nony B Lapeyre J C Ricci S 2019 Prédiction et reconstruction de données hydrauliques par apprentissage machine *unpublished*
[12] Demazels L Le Pape E 2021 Modélisation Hydrologique de la Garonne à Toulouse *unpublished*
[13] Vidal J J Dupouyet J P Murillo T Deltheil T Boignard J P 1998 SOPHIE : Système Ouvert de Prévisions Hydrologiques Informatisé avec Expertise *Journées de l'Hydraulique* pp 389-398
[14] Pedregosa *et al.* 2011 Scikit-learn: Machine Learning in Python, *JMLR 12* pp 2825-2830.
[15] Chollet *et al*. 2015 Keras https://keras.io
[16] Friedman J H 2001 Greedy function approximation: a gradient boosting machine. *Annals of statistics* pp 1189-1232
[17] Gull A Pal S 2017 Deep Learning with Keras *Packt Publishing Ltd.*
[18] Bergstra J Bengio Y 2012 Random search for hyper-parameter optimization *The Journal of Machine Learning Research*