Reproducing human decisions in reservoir management: the case of lake Lugano

G. Corani[1] , A.E. Rizzoli[1] , A. Salvetti[2] and M. Zaffalon[1]

(1) IDSIA (Istituto Dalle Molle Intelligenza Artificiale), 6928 Manno-Lugano, Switzerland, e-mail: giorgio@idsia.ch

(2)Dipartimento del Territorio Cantone Ticino, Ufficio dei corsi d'acqua, 6500 Bellinzona, Switzerland, e-mail: andrea.salvetti@ti.ch
and
Politecnico di Milano, Dipartimento di Elettronica e Informazione, 20122 Milano, Italy, email: salvetti@elet.polimi,it

Abstract

The objective of this study is to identify a model able to represent the behavior of the historical decision maker (DM) in the management of lake Lugano, during the period 1982–2002. The DM decides every day how much water to release from the lake. We combine hydrological knowledge and machine learning techniques to properly develop the model. As a predictive tool we use lazy learning, namely local linear regression. We setup a daily predictor, which achieves good accuracy, with a mean absolute percentage error around 8.5%. Yet, the behavior of the model is not fully satisfactory during the floods. In fact, from an interview with a domain expert, it appears that the DM can even update the release decision every 6 hours during emergencies. We have therefore developed a refined version of the model, which works with a variable time step: it updates the release decision once a day in normal conditions, and every 6 hours during emergencies. This turns out to be a sensible choice, as the error during emergencies (which represent about 5% of the data set) decreases from 9 to 3 m^3/sec.

1. Introduction

Reservoirs play a major role in the storage of water for a number of purposes, ranging from irrigation of agricultural fields, to drinking water supply, hydroelectric power production, and even recreational uses. Their management is often performed by human decision makers, who are able to mix various information sources, both qualitative and quantitative, to make daily release decisions. A number of studies have shown that reservoir management can be mathematically modeled and the problem can be formulated as a multi-objective optimization problem (Soncini et al., 2007). Yet, the design of complex and large reservoir systems often imposes that different reservoirs have different types of regulation: a major reservoir might be ruled by an optimized policy, while smaller ones are still regulated by hand. In the computer simulation of the reservoir network it becomes essential to be able to reproduce the behavior of the human decision makers, in order to effectively and rapidly test the system under various scenarios, which is particularly important in the context of model validation and also in assessment and participatory decision making studies.

The objective of this study is to identify a model able to represent the behavior of the historical decision maker (DM) in the management of lake Lugano during the period 1982–2002; we have not found significant literature on similar problems, despite the huge number of works dealing with river flow forecasting and optimal reservoir management.

The DM decides every day how much water to release from the lake, on the basis of different pieces of hydrological information such as lake level, rainfall and inflow. In order to develop the model, we combine existing hydrological knowledge with knowledge acquired by machine learning: we use machine learning to learn the model from the data and hydrological knowledge to choose sensible inputs and to refine the model. As regression algorithm, we use *lazy learning* (LL) (Bontempi et al., 2001).

We start by developing a LL model with daily time step. A rough predictor of the DM decision is to set today's release equal to yesterday's release plus a correction term, proportional to the variation of the lake level over the last 24 hours; we denote the release computed in this way as naïve release. However, the behavior of the DM becomes different from the naïve release in case of significant rainfall; to properly mimic the DM's behavior, we setup a daily LL model, which has as input the naïve release and the past two daily measures of rainfall. The daily LL model achieves a good accuracy: the mean absolute error (*mae*) is 1.9 m^3/sec and the mean absolute percentage error (*mape*) is 8.6% (1.9/22 = 0.086); the correlation between true and predicted values is close to 1. Yet, the behavior of the model is not fully satisfactory during the floods. Analyzing the problem with the domain expert, it was pointed out that the DM might also update the release decision every 6 hours during emergencies.

We therefore refine the model, by introducing a variable time step; the model takes the release decision once a day in normal conditions, and every 6 hours during the emergencies. The conditions, which determine whether we are in an ordinary situation or in an emergency, are designed according to the suggestions of the domain expert and are outlined in Section 5. The number of emergency steps turns out to be around 5% of the data set; this prevents a major improvement of *mae* and *mape* from being achieved, as they are averaged over all the time steps. To analyze the effectiveness of the model with variable time step, we rather have to carefully examine emergencies. During emergencies, *mae* is about 9 m^3/sec for the daily predictor and only 3 m^3/sec for the model with variable step; correspondingly, since the average release during emergencies is about 100 m^3/sec, *mape* is respectively 9% and 3%. The introduction of the variable time step is therefore clearly beneficial for the accuracy of the model.

The paper is organized as follows: in Section 2 we describe the case study, in Section 3 the LL algorithm, in Sections 4 and 5 the daily predictor and the predictor with variable time step.

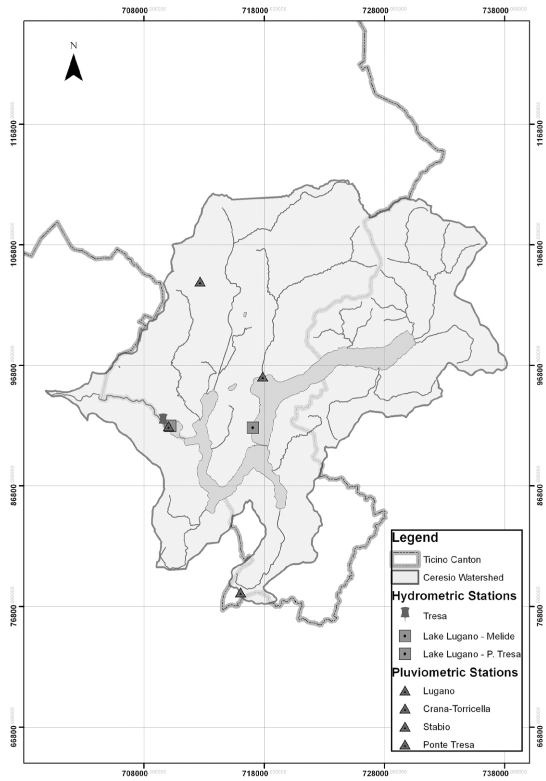
2. The case study

The catchment of Lake Lugano, shown in Fig. 1, has an area of about 615 km^2, 40% of which are in Italian territory while the reminder lies in Switzerland. The surface area of the lake is about 49 km^2 and the outlet is at Ponte Tresa, where the river Tresa flows for 13 km along the Italian-Swiss border and finally into Lake Maggiore.

The ratio between the catchment area and that of the lake is 13 to 1, similar to Lake Geneva (14 to 1) and smaller compared to the ratio of Lake Maggiore (31 to 1).

The catchment has a typical Prealpine pattern, characterized by quite steeply sloping mountainsides in the Northern part of the basin and by more gentle hills in the Central and Southern part.

The average elevation of the catchment is about 1000 m a.s.l., a maximum elevation of 2150 m a.s.l., and the average level of the lake is about 270.50 m a.s.l. The catchment can be subdivided into several tributary



subcatchments: the principal ones are the Vedeggio, the Cassarate, the Magliasina, and the Laveggio.

The rainfall regime is typical of South Alpine zones and, in fact, the trajectory of the (median) inflow to the lake, as depicted in Fig. 2 for the years 1982-2002 has an absolute minimum in winter and two peaks in autumn and late spring. The precipitation amount is relatively high (about 2000 mm per year) and not uniformly distributed during the year.

Fig. 1: The Lake Lugano watershed and the main rainfall and hydrometric gauging stations.

All the tributaries are characterized by a torrential runoff regime and the time response is quite short (6-12 hours); therefore, floods caused by intense precipitations over the whole basin are fast and sudden, particularly if they come after a wet period, when the soil is saturated.

In natural regime, the dynamics of the level and the outflow are driven exclusively by the inflow, with a short delay between inflow and outflow provoked by the retention capacity of the lake.

The construction of the dam at Ponte Tresa was finished in 1962 and the regulation started in 1963, with the goal of reducing flood events in

Lugano and, parallel, to stabilize the outflow from the lake, in order to increase the energy production of the ENEL hydropower plant constructed 1933 along the Tresa in the Italian territory.

The international Regulation License, signed in 1955 between Italy and Switzerland in Lugano, establishes a one-to-one relationship between lake levels at Ponte Tresa and releases in the Tresa river. The Regulation Licence is time-dependent and takes into account the pattern of the median annual inflow, but, being the regulation time step (24 hours) longer that the response time of the tributaries (6-12 hours), intense precipitation events can significantly increase the level of the lake in a few hours, also by strictly applying the regulation rule.

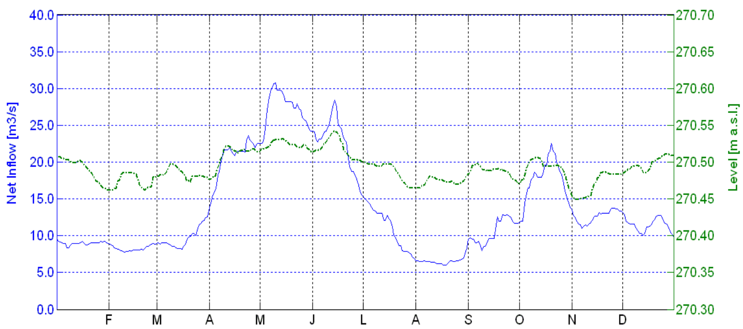


Fig. 2: The median annual trajectory of the median inflow (blue, solid line) and level of Lake Lugano (green, dashed line).

The regulation of the lake has produced a reduction of the number and intensity of the flood events and a smoothing of the duration curve of the outflow from the lake. The number of days with release between the minimum and the maximum turbine capacity of the power plant increased from 190 days in the natural regime up to 250 days in the regulated regime.

The DM devoted particular attention to the flood protection objective and he tried to “improve” the Regulation License, mainly before and during intense and sudden flood events, by exploiting more information than only the lake level as stated by the official License (see Introduction). In this respect, information about the inflow amount and the rainfall (past and forecasted) over the whole basin was surely used by the DM in the actual policy implementation. The effect of the regulation can be seen in Fig. 2, which shows the median inflows and lake levels over the period 1982-2002.

Hourly measurements of the lake level are collected in two sites, at Ponte Tresa, close to the regulation dam, and at Melide, whose level is more representative for the level condition in Lugano; the release is directly measured at the regulation gates and, additionally, at an automatic gauging station along the Tresa, about 1 km downstream.

Daily rainfall measurements are collected at 4 locations, two of which (Stabio and Lugano) are actually automatic stations with a 10 minutes time resolution. The whole rainfall dataset was used for daily model, and the information of the two automatic gauging stations was taken into account in the refinement of the model with variable step size (see Section 5 for details).

3. Lazy Learning (LL)

In local modeling one renounces to a complete description of the target function, to instead approximate it only in the neighborhood of certain points of interest (*query-points*). In particular, LL is a local linear model; in fact, thanks to locality, LL makes it possible to address non-linear problems via linear methods.

Let us assume we want to predict Y on the basis of X_1 and X_2. At time t, we read $x_1(t)$ and $x_2(t)$; the vector $[x_1(t)\ x_2(t)]$ constitutes the query-point. LL builds a linear approximation of the target function around the query-point as follows: (i) it looks for the instances $[x_1\ x_2]$ of the training set that minimize the distance from the query-point; (ii) it fits a linear regressor by considering only the k closest instances and then (iii) it returns the forecast. In fact, a new local regressor is generated for each query point, that is, the local approximation of the target function is re-computed at each query-point.

The number k of instances used to identify the local model is referred to as bandwidth, and determines the size of the region spanned by the linear approximation; to correctly identify the bandwidth is of major importance in local modeling. The LL variant designed by Birattari et al. (1999) optimizes the bandwidth query by query as follows: on each query, it identifies a set of local linear regressors, having bandwidths comprised between k_{min} and k_{max} (in this application, after some trial and error, we have set k_{min}=50 and k_{max}=200; therefore, 151 local regressor are identified for each query); then the best local regressor is chosen via a model selection procedure and it is eventually used to return the prediction. In this way, the bandwidth is managed adaptively. Nevertheless, the whole algorithm is computationally efficient, since both the parameters of the local regressors and their cross-validation errors are computed recursively.

After having returned the prediction, LL discards the local regressor and keeps in memory the training set in order to perform the local learning on future queries; LL is thus a memory-based algorithm. On the one hand, this requires additional memory compared to global models which, after having been trained, only store the estimated parameters; on the other hand, this allows LL to be easily kept update; it is enough to add the new instances to the memory of LL.

LL can be thought of as not requiring an actual training stage, thus enabling a nice time saving compared to neural networks (or global models in general); in fact, it is lazy as it defers the estimation of the regressor until a prediction is required, namely until a query-point is provided. However, LL does perform local learning in order to compute the prediction; it is hence slower than global models in returning the prediction. This is not a problem however for the present case study, as LL takes only a few seconds to compute a few thousands predictions on a standard PC; this result is due both to the small set of inputs (which speed up distance computation) and to the efficient implementation by Birattari et al. (1999) .

A further nice feature of LL (and of local models in general) is how it deals with the distribution P(**X**) of the inputs. In global modeling, one tries to minimize an objective function which spans the whole domain of **X**, thus biasing the global approximator towards those regions where P(**X**) is larger. In local modeling, on the contrary, each prediction problem is solved independently of any other one, thus bypassing the problems originated by negative interferences between different regions of the input space (Bontempi et al., 2001).

The LL package used in this work has been already shown to achieve high accuracy in non-linear modeling problems, such as prediction of air pollution (Corani, 2005) and prediction of temperature of power components (Villacci et al., 2005).

4. Daily predictor

We want to predict the release decision *rel(t)*; it is taken at 7 a.m. of day *t* and it is measured as the average outflow (m^3/sec) observed between 7 a.m. of day *t* and 7 a.m. of day *t+1*. As input variables, we consider the variation of lake level during the last 24 hours ($\Delta h(t) = h(t)_{7a.m.} - h(t-1)_{7a.m.}$) and the two last measures of daily rainfall, *rain(t–1)* and *rain(t)*. More precisely, *rain(t)* denotes the total rainfall measured between 6 a.m. of day *t–1* and 6 a.m. of day *t*.

The rainfall is measured in four different places; for each time instant, we aggregate these four measures into a single one via Thiessen polygons (also referred to as Voronoi diagrams). In this way, we get an estimate of the rainfall on the whole catchment, starting from several spatially distributed measures.

Two remarks about the notation: for all the variables apart from the release, the index *t* denotes the day on which the measure is known (see for instance the definition of *rain(t)*); instead, for the release, *t* denotes the time at which the decision is taken (see the definition of *rel(t)*).

An interview with the domain expert points out that in normal conditions the DM roughly follows this rule:

$$rel(t+1) = rel(t) + \Delta h(t)A/\Delta t,$$

where *A* represents the lake surface, i.e. 48.9 km^2; the quantity $\Delta h(t)A$ is then the variation of water stored in the lake (expressed in m^3) during the last day. To practically compute the above formula, it is necessary to convert *A* in m^2 and to assign to Δt the value of the total number of seconds in a day ($\Delta t=3600*24$). The release modeled by the above formula is referred to as rel_{naive}; the rationale behind such criterion is that the DM adjusts the release proportionally to Δh, trying to keep the lake level constant. In fact, if the lake level has been constant over the last day ($\Delta h=0$), we have $rel(t+1) = rel(t)$. This is a good approximation of the DM policy in normal conditions; in fact, the naïve rule has a good predictive power, with a *mae* of 2.3 m^3/sec on the test set. However, the DM decision does not match rel_{naive} when the last measures of rainfall are above 20 or 30 mm/day. This can be seen from Table 1, which reports the *mae* for different ranges of *rain(t)*.

In order to improve the naïve rule, we identify a predictor of type $rel(t)= f(rain(t), rain(t-1), rel_{naive})$. Note that we use directly rel_{naive} as input for the model, as this simplifies the learning task. We have tested linear regressors, regression trees and LL; in the following we focus on LL only, as it has shown the best performance.

After some trial and error, we have identified the following piecewise structure: if $rain(t) <40$ mm, *rel(t)* is predicted as $LL(rel_{naive}(t), rain(t), rain(t-1))$; if $rain(t) >40$ mm, *rel(t)* is predicted as $LL(rel_{naive}(t), rain(t+1), rain(t), rain(t-1))$. The rationale is that, if the last rainfall measure is high, the DM also considers the rainfall forecast for the day ahead; however, such information is unavailable to us and therefore we use in its place the historically observed *rain(t+1)*. Summing up, the model uses two or three inputs, depending on the value of *rain(t)*.

Table 1. Performance of naïve release and LL for different ranges of rainfall.

rain(t) (*mm/day*)	*Avg. release* (m^3/*sec*)	*Instances* (*%*)	rel_{naive}		*Piecewise LL*	
			mae	*mbe*	*mae*	*mbe*
0–20	10.1	68.7%	0.7	0.2	0.7	0.3
20–40	27.5	16.6%	3.5	0.6	3.0	0.1
>40	75.9	14.7%	8.5	–1.7	5.9	–1.5
AVG	22.7		2.3	0.0	1.9	0.0

A specific analysis has been then carried out to better understand the negative bias present for *rain(t) >40*, even for the piecewise LL. There is in particular a small number of instances over which the model makes large underestimation errors, corresponding to the two main flood events occurred in the considered period. An interview with the domain experts points out that the DM can update the release every 6 during an emergency; this cannot be captured by a daily model. In order to deal with this issue, we develop an improved model; it updates the release once a day in normal conditions and every 6 hours during the emergencies.

5. Model with variable step size

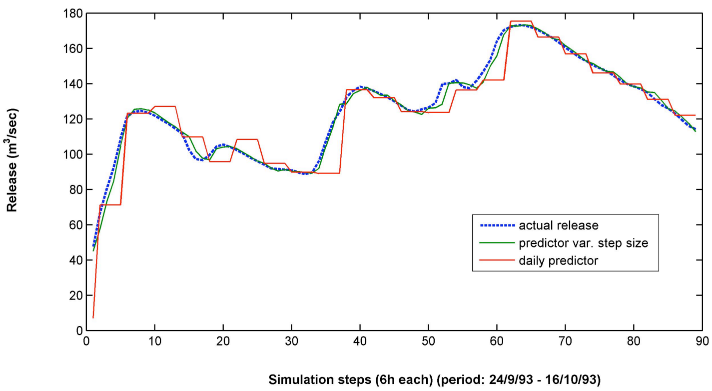
The model with variable step size works on a daily basis, using the predictor of Section 4; yet, during the emergencies an emergency predictor is used to update the release every 6 hours.

Note that in the following, as we deal with the emergency predictor, the time interval between *t* and *t+1* is of 6 hours. On the basis of the recommendations of the domain expert, we define the emergency conditions as follows:

- emergency starts if the lake level *h(t)* has increased more than 3 cm. during the last 6 hours AND *h(t) >270.6 m*;

- emergency is over when *h(t)* $<270.75m$ AND $-2cm < \Delta h < 0cm$ AND *rain(next48h)*[1] $< 30mm$.

The emergency predictor is again a LL model, for which we consider the following input variables: *h(t)*; *inf(t)* − *rel(t−1)*, i.e., the difference between inflow and release over the last 6 hours; the rainfall forecasts for the next 12, 24 and 48 hours. This set of attributes leads to a *mae* of about $10m^3$ /sec, as evaluated by cross-validation on the emergency instances of the period 1982–1992 (note that we do not use at this stage the test data, as we are still trying to identify the optimal structure for the predictor). However, it turns out that the parameter of the rainfall forecast for the next 24 hours fluctuates around zero; we have assessed this by inspecting the parameters of the local regressors estimated on the different query-points. In fact, the 24-hours forecast can be (approximately) seen as a weighted average of the 12 and the 48-hours forecast; in this sense, it mostly provides redundant information to the model. We therefore remove this input and then we insert as further input also the naïve release (computed on 6-hours time steps). The combined effects of these two adjustments leads to a much smaller *mae* (about $3m^3$/sec), as measured by cross-validation on the emergency instances of the training set.



[1] The term *rain(next48h)* denotes the rainfall forecast for the next 48 hours; note that however we will work with the actual rainfall, as the forecast are no longer available.

Fig. 3. Comparison between the daily predictor and the predictor with variable step size on a flood occurred in 1993.

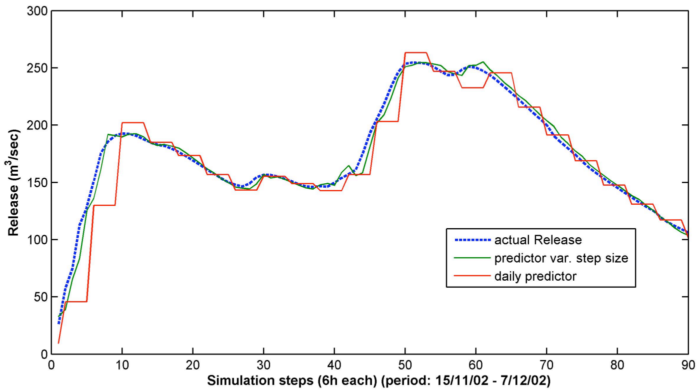


Fig. 4. Comparison between the daily predictor and the predictor with variable step size on a flood occurred in 2002.

At this point, the structure of the model is identified; we then validate the model on the test set (1993–2002). In principle, no large improvements can be expected of the average indicators compared to the daily predictor, since the number of emergency instances is limited (about 5% of the data set) and the average indicators are already fairly good. However, a major improvement can be specifically expected during the emergencies. During the emergencies, the *mae* is reduced to 2.9 m^3/sec, compared to more than 9 m^3/sec of the daily predictor. Therefore, the adoption of the 6-hours predictors allows reducing the *mae* by a factor of three during the floods. There is also a small improvement of the overall *mae*. We can eventually conclude that the switch mechanism allows us to largely decrease the error during the emergencies. The plots show in Fig. 3 and 4 show that the 6-hours predictor is truly suitable to follow the DM decision during the flood.

Conclusions

In this study we designed an algorithm able to reproduce the behavior of a human decision maker in the management of a water reservoir. The algorithm is based on both mechanistic hydrological knowledge and on knowledge learnt from data and observations by means of machine learning techniques. We used machine learning to learn the model from the data and hydrological knowledge to refine the model by selecting the variable step size and in the design of a switch able to select in which state the model should be: normal or emergency mode, the latter to be used during extreme rainfall events.

The integration of machine learning techniques with hydrological knowledge has proven to be a successful approach to the problem at hand, enabling us to achieve a high accuracy in the reproduction of the human decision, by a threefold reduction of the mean average error with respect to a data driven only approach. In relative terms, the mean absolute percentage error is limited to 3% on the validation data set.

This algorithm is expected to be a useful component for the simulation of large reservoir networks, where an automatic mechanism to simulate the human decision making process is needed.

Acknowledgements

Work partially supported by the Swiss NSF grants n. 200020-116674/1, n. 200020-121785/1 and 200021-118071/1.

References

Birattari M., Bontempi G., Bersini H. (1999). Lazy learning meets the recursive least squares algorithm In: M. S. Kearns, S. A. Solla, and D. A. Cohn (eds), Neural Information Processing Systems 11, MIT Press, Cambridge, MA, pp. 375-381.

Bontempi G., Bersini H., Birattari M. (2001). The local paradigm for modeling and control: from neuro-fuzzy to lazy learning. Fuzzy Sets and Systems 121/1: 59-72.

Corani, G. (2005) Air quality prediction in Milan: feed-forward neural networks, pruned neural networks and lazy learning. Ecological Modeling 185: 513-529.

Soncini-Sessa R., Castelletti A., Weber E. (2007). *Integrated and Participatory Water Resources Management*. Elsevier, Amsterdam.

Villacci D., Bontempi G., Vaccaro A., Birattari M. (2005). The role of learning methods in the dynamic assessment of power components loading capability. Industrial Electronics, IEEE Transactions on 52: 280-290