Dynamic emulation modelling of a 1D hydrodynamic-ecological model: Tono Dam case study

A. Castelletti S. Galelli M. Restelli R. Soncini-Sessa

Abstract: Optimal management of water resources systems is often limited by the high computational burden associated to the adoption of process-based models. In this paper we propose a procedural approach for the construction of simple, computationally efficient models that emulate the main dynamics of the original process-based model, but with reduced computational requirements, and that can thus be employed for the resolution of optimal control problems. The core of the procedure is the novel Iterative Feature Ranking algorithm, through which the most relevant variables in explaining the objective function of the control problem are selected among the large set of candidate variables associated with the original process-based model. The final emulation model is then identified using Extremely Randomized Trees. The approach is demonstrated on a real-world case study (Tono Dam, JP).

Keywords: Dynamic models; System identification; Nonlinear systems; Redundance reduction; Data reduction; Machine learning; Environmental engineering.

1. INTRODUCTION

In the water quality sector Process-Based (PB) models are widely adopted to describe the hydro-biological processes of water bodies and reservoirs. However, such modelling approach comes to a price, since the computational requirement of these large models, in terms of CPU, memory and disks usage, is considerably high. This prevents their employment for optimal control problems, which typically require hundreds or thousands of model runs.

A common and effective approach to overcome this limitation is to perform a reduction of the original PB model by identifying a simplified, computational efficient emulation model, which mimics the dynamical behavior of the underlying system under the effect of external drivers. An emulation modelling problem can be tackled with different techniques, distinguishable into three main methodological groups: i) techniques based on the heuristic simplification of the PB model structure, via parametric sensitivity and removal of variables/feedbacks mechanisms [Crout et al., 2009]. The adoption of these techniques is typically limited to domain experts, as a certain amount of user interaction is required; ii) techniques that reduce the PB model dimensionality by projecting the original systems of state transition equations into lower-dimensional subspaces [Antoulas et al., 2001]. These techniques are available only for linear and weakly non-linear systems, while theory is still under development for non-linear systems; iii) The last group of techniques comprises all the methods based on the identification of an emulation model via processing of a data-set appropriately generated from the original PB model [Ratto et al., 2007]. These are the techniques that most attracted researchers from the water resources community (see, for example, Galelli et al. [2010], and references therein), where PB models are generally spatially-distributed, highly non-linear and characterized by complex structures.

The purpose of this paper is to propose a six-steps procedure for the identification of emulation models in a data-driven fashion and to introduce the novel Iterative Feature Ranking (IFR) algorithm. The IFR algorithm represents the core mechanism of the procedure, through which the input/state variables to the emulation model are automatically selected among the large set of candidate input/state variables of the original model. The advantages of the proposed procedure and the IFR algorithm are shown for a real-world case study, the reduction of DYRESM-CAEDYM, a 1D coupled hydrodynamic-ecological model describing the hydro-biological processes occurring in Tono Dam (JP).

The paper is organized as follows. The next Section presents the procedure for the identification of emulation models, while Section 3 describes the IFR algorithm. Section 4 provides a brief description of the case study and the related management problem. Section 5 is completely devoted to the emulation modelling problem, describing the generation of the data via simulation with the PB model, the selection of the most relevant input variables and the model cross-validation. Further discussion and concluding remarks are given in Section 6.

2. EMULATION MODELLING

Given a PB model with state, exogenous driver, control and output dimensionality $N_x$, $N_w$, $N_u$ and $N_y$, the purpose of an emulation model exercise is to identify a dynamic emulation model in state-space form, with state,
exogenous driver, control and output dimensionality \( \tilde{n}_x \ll N_x, \tilde{n}_w \ll N_w, \tilde{n}_u \ll N_u \) and \( \tilde{n}_y = N_y \), whose output \( \tilde{y}_t \) accurately reproduces the PB model output \( Y_t \) (i.e. \( \tilde{y}_t \sim Y_t \)): namely

\[
\begin{align*}
\tilde{x}_{t+1} &= \tilde{f}_t(\tilde{x}_t, \tilde{w}_t, \tilde{u}_t) \quad (1a) \\
\tilde{y}_t &= \tilde{h}_t(\tilde{x}_t, \tilde{w}_t, \tilde{u}_t) \quad (1b)
\end{align*}
\]

where \( \tilde{x}_t, \tilde{w}_t, \) and \( \tilde{u}_t \) are the reduced state, exogenous driver and control vectors, \( \tilde{f}_t(\cdot) \) a non-linear, time-variant, vector function modelling the dynamics of \( \tilde{x}_t \), and \( \tilde{h}_t(\cdot) \) a non-linear, time-variant, output transformation function. Such identification problem can rely on the employment of a discrete data-set \( \mathcal{F} \) of tuples \( \{t, X_t, W_t, U_t, Y_t, X_{t+1}\} \) (with \( t \in [1, H] \)) obtained via simulation of the PB model on a given horizon \( H \). The emulation model can be identified by employing the following six-steps procedure (for further details, see Galelli [2010]).

- **Step 0. Problem conceptualization.** As the emulation model must be used for the resolution of optimal control problems, it is assumed that the design objectives defining the problem and the associated step-indices have already been defined. The step-indices are the natural output \( Y_t \) and \( \tilde{y}_t \) of both PB and emulation model, as they represent the most general piece of information required by any optimization algorithm.

- **Step 1. Design Of Experiments (DOE) and simulation runs.** The purpose of the DOE is to design a sequence of simulation experiments to be performed with the PB model to generate the data-set \( \mathcal{F} \). The DOE is a sampling problem in the space of \( W_t \) and \( U_t \). A general framework for the generation of these trajectories is not available, and it is common practice to rely on physically-based analysis [Galelli et al., 2010] or pseudo-random generation\(^1\). As PB models are generally run with a small simulation time-step to ensure numerical accuracy, a sampling interval can be defined, in order to reduce the number of samples that will compose the discrete data-set \( \mathcal{F} \). This sampling interval will correspond to the emulation model time-step. According to the DOE outcome, the PB model simulations are run and the data composing the discrete data-set \( \mathcal{F} \) are collected with the sampling interval defined during the DOE.

- **Step 2. Lumping.** Since the core of an emulation modelling problem is to reduce the PB model dimensionality, the most crucial step stands in the selection, from the vectors \( X_t, W_t \) and \( U_t \), of the sub-sets \( \tilde{x}_t, \tilde{w}_t \) and \( \tilde{u}_t \) of variables that will constitute the emulation model. This operation is conceptually well defined, but its practical realization is made difficult by the large dimensionality of vectors \( X_t \) and \( W_t \). In real-world applications \( N_x \) is very large, since the practical usage of PB models often requires to discretize the space domain with a number of cells of the order of \( 10^4 \sim 10^8 \). The values of \( N_w \) are generally smaller (say of the order of \( 10^2 \) when \( W_t \) is distributed in space). For this reason, the purpose of the lumping step is to transform the vectors \( X_t \) and \( W_t \) into the lower-dimensional vectors \( \tilde{x}_t \) and \( \tilde{w}_t \) (with dimensionality \( n_x \ll N_x \) and \( n_w \ll N_w \)), among which to select the elements belonging to \( \tilde{x}_t \) and \( \tilde{w}_t \). As for \( U_t \), a lumping process is typically not required, since it is assumed that control is not distributed in space (and thus \( n_u = N_u \)). Such spatial aggregation can be performed by relying on physical considerations and a priori knowledge, or on formal methods such as feature extraction techniques.

- **Step 3. Reduction.** Based on the information contained in the data-set \( \mathcal{F} \), the purpose of this step is to select, from the vectors \( X_t, W_t \) and \( U_t \), the sub-sets \( \tilde{x}_t, \tilde{w}_t \) and \( \tilde{u}_t \) that will constitute the arguments of the emulation model output transformation and state transition equations. The reduction must be such that the PB model output \( Y_t \) is accurately reproduced, while the emulation model dimensionality \( n_x, n_w \) and \( n_u \) is minimum. Such operation can rely on the adoption of features selection algorithms, which, given a data-set \( \mathcal{F} \) and an output variable \( Y_t \) to be explained, automatically select the most relevant variables among a candidate set (for a survey about these techniques, see Guyon and Elisseeff [2003], Hong et al. [2008]). For each state variable included in the selected subset, the input and state variables selection must be then repeated (assuming as new output the selected state variable), until only disturbances and decisions remain in the subset. Next section is completely devoted to this step, proposing the novel IFR algorithm.

- **Step 4. Model identification.** This is a traditional model identification problem, composed of model selection, parameter estimation (calibration) and validation. As far as the model class \( \tilde{f}_t(\cdot) \) and \( \tilde{h}_t(\cdot) \) of the emulation model is concerned, it is advisable to employ the same class adopted in Step 3. Parameter estimation and validation can be finally performed by adopting suitable calibration/validation algorithms. If the validation performance measures are satisfactory, the model can then be used in the subsequent step.

- **Step 5. Model usage.** Finally, the emulation model is used for the resolution of the optimal control problem for which it was conceived. Eventually, the solution so obtained can be simulated with the PB model, thus enhancing the reliability of the solution itself.

3. EXTRA-TREES - ITERATIVE FEATURE RANKING (ETS-IFR) ALGORITHM

The purpose of Step 3 (Reduction) is to select, among the input/state variables of the PB model, the most relevant variables (features) that will constitute the arguments of the emulation model output transformation function and state transition equation (see eqs. (1)). Features selection algorithms adopted for this task must account for both significance and redundance [Maier et al., 2010]. In other words, they must be able to select only the most

---

\(^1\) Alternatively, when the DOE factors are the PB model parameters, whose value remain constant for the whole simulation run, different statistical techniques (e.g. orthogonal arrays, latin hypercube, etc.) can be employed to sample effectively the space of the factors while accounting for the limited number of simulations that can be run [Kleijnen et al. 2005].
relevant features, while trying to avoid the inclusion of redundant features, which would cause unnecessary model complexity. Literature shows a variety of different feature selection methods, whose characteristics vary with the feature selection problem being considered. This section introduces the IFR algorithm, which, besides providing good performances from a significance and redundancy point of view, shows good capabilities in dealing with the high dimensional data-set characterizing emulation modelling problems.

3.1 Iterative Feature Ranking (IFR) algorithm

The ideal algorithm to be used in the selection of the most relevant features should account for non-linear dependencies and redundancy between features, since the processes simulated by environmental PB models are typically non-linear and strongly correlated. Moreover, such algorithm must be computationally efficient in dealing with the usually large number of candidate features. To fulfill these requirements, we developed a model-free, forward-selection algorithm, named IFR algorithm. Given the output $r_t$ and the vector $z_t$ of candidate features, the IFR algorithm first globally ranks the elements of $z_t$ accordingly to a statistical measure of significance that accounts for non-linear dependencies, and then refines the ranking by evaluating the individual contribution of the features ranked in the first $p$ positions. The most significant feature is then selected and employed as regressor for a pre-defined model class $c(\cdot)$. To account for features redundancy, the algorithm proceeds by repeating these operations on those data that still have to be explained, namely on the residuals of the model built at the previous iteration. The algorithm iterates these operations until the selection of new features does not further improve the performance of the model being built as measured by some appropriate index.

The IFR algorithm thus requires the selection of an effective statistical measure of significance, which, in turn, influences the choice of the model class $c(\cdot)$. The parameters to be specified are thus $p$ and the tolerance $\epsilon$ used to terminate the algorithm. A tabular version of the IFR algorithm is given in Appendix A, while, for further details, the reader is referred to Galelli [2010], Castelletti et al. [2011]. As for the model class $c(\cdot)$ and the statistical measure of significance, we propose to employ a class of tree-based methods, named Extremely Randomized Trees (Extra-Trees; Geurts et al. [2006]), and an Extra-Trees based ranking procedure. Their characteristics and advantages are shown in the next two Sections.

3.2 Extremely Randomized Trees (Extra-Trees)

Tree-based methods are non-parametric supervised learning methods that can provide several desirable features in regression problems, as modelling flexibility, computational efficiency, good accuracy and interpretability. They are all based on the idea of decision tree models, which are tree-like structures representing a cascade of rules leading to numerical values [Breiman et al., 1984]. These structures, composed of decision nodes, branches and leaves, are obtained by partitioning, according to a certain splitting criterion, the set of the input variables into a series of subsets, until either the numerical values belonging to each

sub-set vary just slightly or only few elements remain. When the splitting process is over, the branches represent the hierarchical structure of the sub-set partitions, while the leaves are the finest sub-sets associated to the terminal branches. Each leaf is finally associated with a numerical value. Tree-based methods include both deterministic (e.g. classification and regression trees; Breiman et al. [1984]) and randomized methods (e.g. tree bagging and random forests; Breiman [2001]), which differ each other by the splitting criterion, the termination test they adopt, the number of trees they grow, and the rule they use to associate a numerical value to each leaf.

Extra-Trees belong to the family of randomized methods, since they randomize the selection of the input variable and its corresponding value (i.e. cut-direction and cut-point) when splitting a node. In order to compensate for randomization, the building algorithm grows ensembles of $M$ trees. Nodes are split using the following rule: $K$ alternative cut-directions are randomly selected and, for each one, a random cut-point is chosen; a score is then associated to each cut-direction and the one maximizing the score is adopted to split the node. The algorithm stops partitioning a node if its cardinality is smaller than $n_{\text{min}}$, and the node is therefore a leaf. A value is assigned to each leaf, obtained as the average of the outputs associated to the inputs falling in that leaf. The estimates produced by the $M$ trees are finally aggregated with arithmetic average. The rationale behind this approach is that the combined usage of randomization and ensemble averaging provides more effective variance reduction than other randomization methods, while minimizing the bias of the final estimate [Geurts et al., 2006]. Extra-Trees are thus characterized by three parameters (i.e. $K$, $n_{\text{min}}$ and $M$), whose value can be fixed on the basis of empirical evaluations:

- $K$, the number of alternative cut-directions, can be chosen in the interval $[1, \ldots, n]$, where $n$ is the number of inputs (cut-directions). When $K$ is equal to $n$, the choice of the cut-direction is not randomized and the randomization acts only through the choice of the cut-point. On the contrary, low values of $K$ increase the randomization of the trees and weaken the dependence of their structure on the output of the training data-set. Geurts et al. [2006] have empirically demonstrated that, for regression problems, the optimal default value for $K$ is $n$.

- $n_{\text{min}}$, the minimum cardinality for splitting a node. Large values of $n_{\text{min}}$ lead to small trees (few leaves), with high bias and small variance. Conversely, low values of $n_{\text{min}}$ lead to fully-grown trees, which may over-fit the data. As for $K$, Geurts et al. [2006] have empirically shown that, although possibly slightly sub-optimal, the default value of $n_{\text{min}}$ equal to 5 appears to be a robust choice in a broad range of typical conditions in regression problems.

- $M$, the number of trees in the forest, influences the strength of the variance reduction and the behavior of the estimation error, which is a decreasing function of $M$ [Breiman, 2001]. The estimation accuracy thus increases with $M$ and the choice of its value depends on a trade-
off between the desired model accuracy and available computing power. The main advantage of Extra-Trees, apart from their computational efficiency and prediction accuracy, is that their building algorithm can be exploited to rank the importance of the n input variables in explaining the output behavior and then identify the most relevant variables among n candidate inputs.

3.3 Feature ranking

Feature ranking based on Extra-Trees (see Fonteneau et al. [2008], and references therein) is based on the idea of scoring each input variable by estimating the variance reduction it can be associated with by propagating the training data-set over the M different tree structures composing the ensemble. More precisely, let’s consider a regression problem with an output variable r, n input variables \( \{z_1, z_2, \ldots, z_n\} \) and a training data-set \( S \), composed of N input-output observations. The relevance \( G(z_i) \) of each input variable \( z_i \) in explaining the output \( r \) can be evaluated as follows

\[
G(z_i) = \frac{\sum_{\tau=1}^{M} \Omega \delta(\nu_j, z_i) \cdot \Delta_{\text{var}}(\nu_j) | S \rangle}{\sum_{\tau=1}^{M} \sum_{j=1}^{\Omega} \Delta_{\text{var}}(\nu_j) | S \rangle}
\]  

(2)

where \( \nu_j \) is the j-th non-terminal node in the tree \( \tau \), \( \Omega \) is the number of non-terminal nodes in the tree \( \tau \), \( \delta(\nu_j, z_i) \) is equal to 1 if \( z_i \) is used to split the node \( \nu_j \) (and 0 otherwise), \( | S \rangle \) is the number of samples in the considered sub-set \( S \), \( \Delta_{\text{var}}(\nu_j) \) is the variance reduction when splitting the node \( \nu_j \), namely

\[
\Delta_{\text{var}}(\nu_j) = \text{var}\{r | S \rangle - \frac{|S_{i,l}| \text{var}\{r | S_{i,l} \rangle}{|S\rangle} - \frac{|S_{i,r}| \text{var}\{r | S_{i,r} \rangle}{|S\rangle}
\]  

(3)

where the terms \( S_{i,l} \) and \( S_{i,r} \) are the two sub-sets of \( S \) satisfying the conditions \( z_i < s_i \) and \( z_i > s_i \) respectively. The input variables are finally sorted by decreasing values of their importance.

4. TONO DAM CASE STUDY

Tono Dam is being constructed these years at the confluence of Kango and Fukuro rivers and will be completed in 2012. With a height of 75 m, it will form an impounded reservoir of 12.4 \times 10^6 \text{ m}^3 (gross capacity), with a surface area of 0.64 \text{ km}^2 and fed by a 38.1 \text{ km}^2 catchment. The reservoir is being built for multiple purposes, as it will provide water for irrigation to several agricultural districts, feed a small hydropower station, provide industrial and drinking water supply and be used for buffering river floods. A part from these water quantity targets, the management of the reservoir must also satisfy some water quality targets, in order to protect the downstream environment and the irrigated crops during their germination phase. The Japanese legislation suggests indeed to reduce the effect of artificially induced temperature variations, i.e. to keep the outflow temperature as close as possible to the natural inflow temperature. To this purpose, the dam is equipped with a withdrawal intake tower that allows to exploit the natural temperature stratification of the water body by releasing active storage water at different levels. The Selective Withdrawal Structure (SWS) is equipped with a rack of vertically stacked siphons, starting at 18 m from the reservoir bottom.

Tono Dam management problem requires to design a set of Pareto-efficient operating policies, which account for the different management objectives (for further details, see Castelletti et al. [2010]). To this purpose, an optimal control problem has to be formulated and solved, based on the step-indicators that represent the water quantity and quality targets. In this case, considering the methodological focus of this paper, we just consider the water quality step-indicator \( g_t \) that is defined as

\[
g_t = T^\text{out}_t - T^\text{in}_t
\]  

(4)

where \( T^\text{out}_t \) is the average outflow water temperature in the interval \([t-1, t]\), while \( T^\text{in}_t \) is the average temperature of the inflows in the same time interval.

As the management problem requires to account for the spatial dynamics of water quality variables, a 1D coupled hydrodynamic-ecological DYRESM-CAEDYM [Imenito, 2007] model is adopted, whose output variable \( Y_t \) corresponds to the step-indicator \( g_t \). DYNAMIC REServoir Simulation Model (DYRESM) is a one-dimensional hydrodynamics model used for predicting the velocity, temperature and salinity distribution in natural water bodies subjected to external environmental forcing such as wind stress and surface fluxes. The Computational Aquatic Ecosystem Dynamics Model (CAEDYM) consists of a series of mathematical equations representing the major biogeochemical processes influencing water quality, including primary production, secondary production, nutrient and metal cycling, and oxygen dynamics and the movement of sediments.

5. APPLICATION RESULTS

Since the large dimensionality of the PB model DYRESM-CAEDYM precludes the adoption of any optimization algorithm, an emulation model must be identified and calibrated on the data obtained via simulation of the PB model. The target is to reduce as much as possible the number of variables involved in the PB model, while accurately reproducing the step-indicator \( g_t \). To this purpose, the proposed emulation modelling procedure, together with the IFR algorithm, is adopted.

5.1 DOE and simulation runs

With the purpose of exploring as homogeneously as possible the state space of the PB model, 100 pseudo-randomly generated scenarios of the control variable \( u_t \) (i.e. the amount of water withdrawn from six siphons) are combined with the hydro-meteorological data (exogenous driver \( W_t \) measured over the period 1995-2006). For each of the 100 simulation scenarios so obtained, the coupled DYRESM-CAEDYM model is run. The simulated data, sampled with a daily time-step, are finally stored in the
data-set \( F \) of tuples \( \{X_t, W_t, u_t, Y_t, X_{t+1}\} \), with dimensionality \( N_x, N_w, N_u \) and \( N_y \) respectively equal to \( \sim 10^3, 50, 6, \) and 1. The number of collected tuples is equal to \( 4.38 \times 10^5 \).

5.2 Lumping and reduction

The purpose of the lumping step is to transform the state vector \( X_t \) into a lower dimensional vector \( x_t \) (with dimensionality \( n_x \ll N_x \)). This is performed by choosing, on the basis of physical considerations, the state variables that are expected to be significant with respect to the PB model output \( Y_t \): these are the reservoir water temperature and suspended solids concentration at different depths. Three further state variables are considered as components of \( x_t \): the reservoir level \( h_t \) and storage \( s_t \), and the variable \( t_{mod \ T} \), which accounts for the system periodicity by taking value in the range \([1, 365]\). The lumping step gives a reduced state vector \( x_t \) with dimensionality \( n_x \) equal to 19. As for the vectors \( W_t \) and \( u_t \), a transformation is not required, since these variables are not distributed in space. At this stage, the vectors to be considered for the reduction step are \( X_t, W_t \) (\( = W_t \)) and \( u_t \), with dimensionality \( n_x, n_w \) (\( = N_w \)) and \( n_u \) respectively equal to 19, 50 and 6. This gives a total of 75 candidate input variables (features) for describing the dynamics of the output variable \( Y_t \).

The most relevant variables, among \( x_t, w_t \) and \( u_t \), are selected with the IFR algorithm. The Extra-Trees parameters are set following Geurts et al. [2006] indications: the number of alternative cut-directions \( K \) is 75 (i.e. the number of candidate features), the minimum cardinality for splitting a node \( n_{min} \) is 50, while the number of trees in the ensemble is 500. As for the number \( p \) of features singularly evaluated at each iteration and for the algorithm tolerance \( \epsilon \), they are respectively set equal to 5 and 0.

Table 1 reports the results obtained during the features selection process. It can be noticed that the dynamic behaviour of the step-indicator \( g_t \) can be described by selecting eight variables, accounting for the different driving forces that contribute to the step-indicator dynamics (e.g. the temperature \( T_K \) of Kango river or the reservoir water temperature at different depths \( T_{sed} \) and \( T_{aff} \)). In particular, it is interesting to notice that the control decisions taken at -3 and -13 meters \((u_t^{-3} \text{ and } u_t^{-13})\) are very significant: this can be simply explained by considering that these two variables represent the warm and cold water intraflow. Finally, the inclusion of the cloud cover \( cloudt \) does not lead to an increase in the emulation model performances and the features selection process is thus stopped. Among the selected features, the reservoir water level corresponding to the inflow intrusion \( h_t^{in} \), the water temperature at the sediment level \( T_{sed} \) and the water temperature at -7 meters \( T_{aff} \) are state variables, and, as such, they require a dynamic description: the IFR algorithm is then employed to select the most relevant variables in determining their dynamics. This process, which is repeated until only disturbances and decisions remain in the selected sub-sets, finally lead to the selection of 8 state variables \((g_t, h_t^{in}, T_{sed}^{-1}, T_{aff}^{-7}, T_{aff}^{-3}, T_{aff}^{-13}, h_{t_{mod \ T}}\text{, }cloud_{t})\), 7 exogenous drivers \((T_K, NH4_K, T_{air}, P_{bas}, \text{cloud}_{t}, rad_{tot}^{short}, NH4_K)\) and 2 control variables \((u_t^{-3} \text{ and } u_t^{-13})\).

### Table 1. Selected features and corresponding performance of the emulation models iteratively obtained with the IFR algorithm for the case of \( g_t \). State variables are denoted in bold.

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Sel. feature</th>
<th>MISO perf. ((R^2))</th>
<th>(\Delta R^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(T_K)</td>
<td>0.1137</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>(h_t^{in})</td>
<td>0.1238</td>
<td>0.0101</td>
</tr>
<tr>
<td>3</td>
<td>(T_{sed}^{-1})</td>
<td>0.2080</td>
<td>0.0842</td>
</tr>
<tr>
<td>4</td>
<td>(NH4_K)</td>
<td>0.2840</td>
<td>0.0760</td>
</tr>
<tr>
<td>5</td>
<td>(u_t^{-3})</td>
<td>0.3375</td>
<td>0.0535</td>
</tr>
<tr>
<td>6</td>
<td>(u_t^{-13})</td>
<td>0.6360</td>
<td>0.2983</td>
</tr>
<tr>
<td>7</td>
<td>(T_{aff}^{-7})</td>
<td>0.6406</td>
<td>0.0046</td>
</tr>
<tr>
<td>8</td>
<td>(T_{aff}^{-3})</td>
<td>0.6534</td>
<td>0.0128</td>
</tr>
<tr>
<td>9</td>
<td>(cloud_{t})</td>
<td>0.6516</td>
<td>0.0018</td>
</tr>
</tbody>
</table>

### Table 2. Structure and performances \((R^2\) in \(k\)-fold crossvalidation\) of the cascade of emulation models.

<table>
<thead>
<tr>
<th>Output variable</th>
<th>(R^2) ((\pm))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(g_t)</td>
<td>0.6534 ((\pm))</td>
</tr>
<tr>
<td>(h_t^{in})</td>
<td>0.7497 ((\pm))</td>
</tr>
<tr>
<td>(T_{sed}^{-1})</td>
<td>0.9183 ((\pm))</td>
</tr>
<tr>
<td>(T_{aff}^{-3})</td>
<td>0.9039 ((\pm))</td>
</tr>
<tr>
<td>(T_{aff}^{-13})</td>
<td>0.9200 ((\pm))</td>
</tr>
<tr>
<td>(h_{t_{mod \ T}})</td>
<td>0.8920 ((\pm))</td>
</tr>
<tr>
<td>(T_K)</td>
<td>0.9974 ((\pm))</td>
</tr>
</tbody>
</table>

To give consistency to the emulation modelling procedure, an ensemble of Extra-Trees (with the same setting adopted during the features selection process) is employed to model the output \( g_t \), as well as the other selected state variables belonging to \( X_t \). The final structure of the model is thus a cascade of emulation models that is calibrated and validated with a \(k\)-fold cross validation \((\text{with } k = 10)\). Their performances in one-step ahead prediction, which reflects the model capabilities in emulating the PB model behaviour, are reported in Table 2.

A comparison of the trajectories for the output \( g_t \) computed by DYRESM-CAEDYM and the emulator (one-step ahead prediction) is given in Figure 1: the emulator shows good capabilities in approximating \( g_t \) behaviour, apart from the under-estimation of the largest peaks.
6. CONCLUSIONS
The paper presents a procedure for the identification of dynamic emulation models to reduce the dimensionality of the PB models commonly adopted to describe the hydro-biological processes in lake and reservoirs. The procedure, whose core is the Iterative Feature Ranking (IFR) algorithm, is tested on the reduction of the 1D coupled hydrodynamic-ecological model DYRESM-CAEDYM, used to model the conditions to Tono Dam (JP). The emulation model performances are quite satisfactory, and the complexity reduction with respect to the original PB model is remarkable (from $\sim 10^9$ to 8 state variables). These encouraging results suggest that the proposed procedure can be applied to other environmental management problems, involving large models, for which “what-if” analysis over a very small number of alternative decisions is presently the only feasible way of supporting decision-making. Further research will concentrate on the development of a more rigorous DOE and lumping step, and on the assessment of the IFR algorithm performances via comparison with other features selection algorithms. Research effort will finally be devoted to the very scope of this emulation modelling exercise, namely the design of an optimal control policy for Tono Dam.

REFERENCES

Appendix A. IFR ALGORITHM - TABULAR VERSION

Step 0. Set $k = 0$ and $\tilde{z}_t$ as empty vector.

- Rank, in decreasing order, the features in the vector $z_t$ according to their statistical measure of significance in explaining the output $r_t$.

- Select the features $z_{t1},...,z_{tp}$ ranked in the first $p$ positions. For $i = 1,...,p$, identify a model of the form $\hat{r}_{t^k} = c(z_{ti})$ and evaluate its performance $R^i$ in explaining $r_t$.

- Denote as $z_{t^k}$ and $r_{t^k}$ the feature and the estimate of $r_t$ corresponding to the model with the highest performance $R^k$. Store $z_{t^k}$ in $\tilde{z}_t$.

- Compute the residual $r_{t^k} = r_t - r_{t^k}$.

Step 1. Set $k = k + 1$.

- Rank, in decreasing order, the features in the vector $z_t$ according to their statistical measure of significance in explaining the output $e_{tk}^{k-1}$.

- Select the features $z_{t1},...,z_{tp}$ ranked in the first $p$ positions. For $i = 1,...,p$, identify a model of the form $\hat{e}_{tk-1}^{k-1} = c(z_{ti})$ and evaluate its performance $R^i$ in explaining $e_{tk}^{k-1}$.

- Denote as $z_{tk}$ the feature corresponding to the model with the highest performance. Store $z_{tk}$ in $\tilde{z}_t$.

- Identify a model of the form $\hat{r}_{tk} = c(\tilde{z}_t)$ and evaluate its performance $R_k$ in explaining $r_t$.

- Compute the residual $r_{tk} = r_t - r_{tk}$.

Step 2 (termination test). If $(R_k - R_{k-1}) < \epsilon$, the algorithm stops. The selected features are stored in $\tilde{z}_t$, with dimensionality $\tilde{n}_t = k - 1$. Otherwise, return to Step 1.