Control-oriented modeling for model predictive control of water reservoir systems

Raffaele Giuseppe Cestari, Andrea Castelletti, Simone Formentin

[a]*Department of Electronics, Information, and Bioengineering, Politecnico di Milano, Via Giuseppe Ponzio, 34, Milano, 20133, Italy*

Abstract

The problem of selecting a predictive model cannot be separated from its use. When managing a system subject to external disturbances challenging to predict, choosing the prediction model to insert within the Model Predictive Control (MPC) framework depicts a significant obstacle. Focusing solely on identifying the best model within a specific class is not a winning choice. In this work, we demonstrate that for the same model class, the identification through simultaneous fitting of historical data and minimization of a backtesting control cost allows us to improve control performances compared to using all data exclusively for historical time-series fitting. This procedure takes place online, thanks to Bayesian Optimization, which iteratively provides the best prediction model at each instant, bringing the controller closer to the ideal performances. The application under study is the management, based on real data, of the Lake Como basin, Italy. The methodology is particularly promising in the control of dry periods, where the impact of predictions is fundamental in maximizing irrigation releases to the agricultural district.

Keywords: model predictive control, bayesian optimization, hyperparameter tuning, control-oriented prediction model, water resources

1. Problem Statement

The problem of managing water resources is complex for countless reasons. The scale of the system, the difficulty in finding information, and last but not least, the dominant presence of disturbances that play a fundamental role in the performance of the control system represent obstacles that must

be overcome to improve the quality of management of this precious resource: the water. In an increasingly evident context of climate change, the empirical control strategies adopted so far by managers are obsolete. In this framework, relying on previous studies on regulation through Model Predictive Control (MPC) of the Lake Como basin (see Cestari et al. (2022), Cestari et al. (2023)), this work aims to propose a new formulation for the definition of the model for predicting the disturbance entering the system: the inflow to the basin. The inflow is a complex disturbance to predict because it is distributed over the coastal stretch of the basin and is not concentrated in a single point. Furthermore, it depends on large and small-scale hydrometeorological factors (see Faber and Stedinger (2001), Ficchì et al. (2016), Denaro et al. (2017)) and fluctuations in the energy market, as upstream flows may depend on the release policies of hydroelectric plants. Given these considerations, in this work, we propose a methodology that does not focus solely on proposing the best model for predicting inflows and then declining it in a predictive control perspective, but instead using, at each time instant, a prediction model that not only takes into account the historical observations of the disturbance but also includes an evaluation of the performances obtainable in a closed loop. At each time instant, the model that minimizes the combination of the identification cost on the historical dataset (incremented up to the last available data) and the control cost calculated on a backtesting window will be used as a prediction model. Given the heavy non-linearity of this overall cost, we adopted a semi-heuristic black box approach: Bayesian Optimization. This iterative algorithm allows us to calculate the minimum point of a cost function whose analytical expression is unknown, basing the selection of the next minimum candidate on Bayes' probability theory. After a reasonable number of iterations, the algorithm guarantees convergence to (at least local) minima. Many works make use of Bayesian Optimization in predictive control. This ensures the applicability of the methodology. See (Piga et al. (2019), Lucchini et al. (2020), Stenger et al. (2020), Abbracciavento et al. (2023), Le and Malikopoulos (2023), Sorourifar et al. (2023)). The main contributions of the work follow:

1. For the first time, as far as we know, we are using the Bayesian Optimization algorithm to calibrate online the parameters of a prediction model of a predictive controller by simultaneously considering the identification costs on the historical dataset (enlarged by including at each time step the last available datum) and the backtesting performance of

the controller in closed-loop.
2. We demonstrate that, for the same model class, if we consider closed-loop performance as the ultimate aim of the control, the choice of the best model of the disturbance does not concern the best model identified on the data.
3. For the first time, we apply this strategy in water resources management, particularly for regulating the Lake Como basin, Italy, validating the control strategy on a high-fidelity simulator, using 6 years of actual data, and comparing its capabilities against the historical regulation.

The remainder of the paper is as follows. In Section 2 the Bayesian Optimization algorithm is presented. Section 3 shows the prediction model class and the motivations behind the choice. Section 4 describes the control problem solved and the compared control architectures. Section 5 shows the numerical simulations and results, obtained on a high-fidelity simulator and real data. Section 6 depicts the final conclusions.

2. Bayesian Optimization

Bayesian Optimization (BO) is an iterative algorithm that allows us to optimize expensive-to-evaluate functions whose analytical expressions are unknown. In our problem, the objective function J_{BO} depends on the parameters θ of the prediction model.

BO has two main components:

- **Surrogate model**: a statistical representation of the objective function that handles its probability distribution over the objective function space.
- **Acquisition function**: a mathematical function that samples the next point from the parameter space where the objective function will be evaluated.

BO finds the most likely candidate minimum value that J_{BO} can reach over the θ domain at each iteration. Since the objective is unknown and expensive to evaluate, BO finds its best approximation through a step-by-step learning procedure. The initial information required is as follows:

- a training set of observations $(\theta_i, J_{BOi} = F(\theta_i))$, for $i = 1, ..., N_t$. Where $F(\theta)$ is the unknown analytical expression of the objective as function of θ and N_t is the training set (initialization) size.

- a prior probability distribution over the objective function space.
- a likelihood function, expressing our beliefs over the properties of the objective function.

BO algorithm is embedded with a re-shaping procedure that allows it to gather all possible information from the minimum number of observations. According to Bayes's theory, the prior is derived from the collection of observations (e.g., the probability distribution before a new sample is collected). In contrast, the posterior distribution is updated after a new sample is selected, and it becomes, iteratively, the prior for the following iteration. Thus, this step-by-step learning procedure involves re-shaping the prior distribution each time a new observation is added to the training set. The acquisition function and its maximization manage the selection of the following sample in the parameter space. The acquisition function looks at the shape of the posterior distribution over the objective function space and decides which is the most promising point with an optimization procedure (e.g., the one maximizing the probability). That point is the best candidate for being at least a local minima for the cost function in the parameter space. After a satisfactory number of iterations K, whose value must be tuned carefully, looking at the convergence rate of the objective, the algorithm can be stopped, returning the desired most likely optimal parameters. BO algorithm is implemented in several programming languages. In this work, we used the Statistics and Machine Learning Matlab toolbox. Entering into the BO algorithm's details is out of this paper's scope. For additional information, refer to Lévesque et al. (2016), Hutter et al. (2019), Wu et al. (2019), Victoria and Maragatham (2021). Figure 1 shows the cost trend as the iterations of the BO increase. A "knee" followed by a plateau is observed between 5 and 15 iterations in different tests. For this reason, we select the maximum number of iterations of the BO algorithm as $K = 30$. The numerical expression of the cost function is introduced in Section 4, together with the control problem.

3. Prediction Model

Our choice of the prediction model class is based on two main aspects:

1. The model must have reduced complexity (the number of parameters must be 10-20 units). This requirement is due to the limited capability of online calibration of many parameters using Bayesian Optimization.

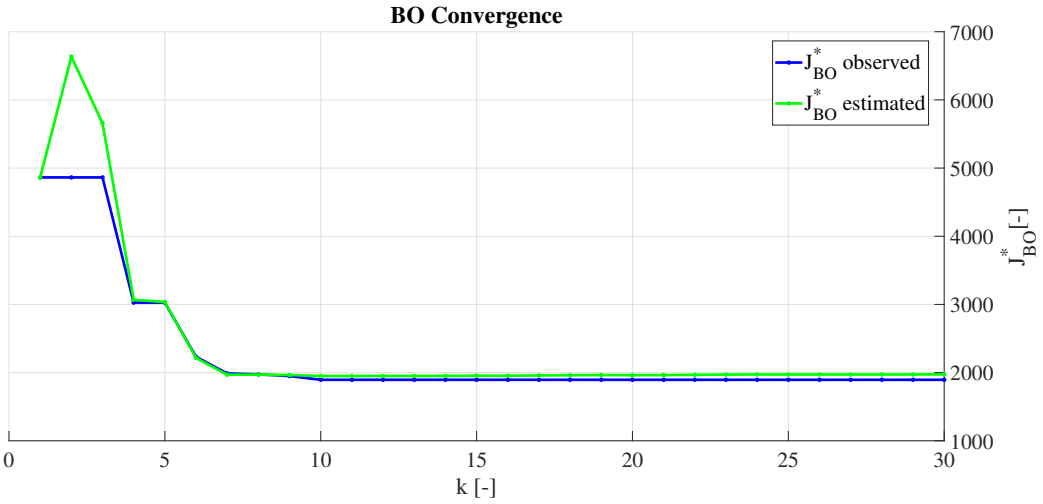

Figure 1: Bayesian Optimization Algorithm Convergence

Indeed, in the literature, it is indicated to keep the number of parameters limited for the correct effectiveness of the Bayesian Optimization algorithm, see Shahriari et al. (2015), Dewancker et al. (2016).

2. The model must be recognized in the literature as usable in predicting inflows.

According to the literature review Mosavi et al. (2018), the class of ARIMA models represents the winning choice. This ensures, at the same time, a solid application in the literature and keeps the number of parameters limited. Given the available historical dataset (January 1, 1946 - December 31, 2019), we performed a grid-search procedure to test the order of the best-performing ARIMA model. The training data go from January 1, 1946 to December 30, 2004. The validation data go from December 31, 2004, to December 31, 2019. The orders covered by grid-search go from ARIMA(0,0,0) to ARIMA(4,4,4). The best model for this class (maximum R^2) is the ARIMA(3,2,4). To further validate the selection, we reduced the size of the training set to 10 and 5 years of training, respectively. This verification was carried out to consider the possible effect of the non-stationarity of the inflow time-series (which would make the most remote data obsolete). These checks confirmed the ARIMA(3,2,4) choice. In Figure 2 we show the historical inflow, ARIMA(3,2,4), and cyclo-stationary average (e.g. the average inflow on each day of each year available) on the validation set. To further ensure the validity of the selected model,

H	10	neurons per layer
ν	0.001	learning rate
μ	0.0001	decay rate
B	256	batch size

Table 1: LSTM setting

Model	R^2
ARIMA(3,2,4)	0.62
LSTM	0.63

Table 2: R^2 prediction models

we tested a Long short-term Memory (LSTM) neural network (see Sangiorgio and Dercole (2020), Sangiorgio et al. (2021)), a type of recurrent neural network aiming at solving the vanishing gradient problem and particularly suited for time-series forecasting, see Sagheer and Kotb (2019), on the same data in the training configuration without teacher forcing (which involves using 2 LSTMs with an additional dense layer). The optimizer used is ADAM, and the maximum number of epochs is 2000. In Table 1 the LSTM setting, obtained after a grid-search procedure, is shown. Both strategies involve the use of a 1-step ahead predictor. Multi-step ahead prediction is obtained in simulation starting from the 1-step ahead predictor. In Table 2, we show how the values of the coefficient of determination R^2 for the two models, one with low complexity (and reduced number of parameters), ARIMA(3,2,4), and the other with high complexity, LSTM, are close. The increase in the complexity of the model allows us to earn only 0.01. This validates the choice of the ARIMA(3,2,4) model that will be used in the predictive control framework. For additional comparison between ARIMA and LSTM models please refer to Elsaraiti and Merabet (2021). Here below we recall briefly the ARIMA(p,d,q) one-step ahead predictor.

$$\Delta\hat{Y}_{t+1} = \phi_1\Delta Y_t + \phi_2\Delta Y_{t-1} + \ldots + \phi_p\Delta Y_{t-p+1} + \theta_1 e_t + \theta_2 e_{t-1} + \ldots + \theta_q e_{t-q+1} \quad (1)$$

Y_t is the time series data, ΔY_t is the differenced series, p is the order of the autoregressive (AR) component, d is the degree of differencing (I), q is the order of the moving average (MA) component, ϕ_i are the autoregressive coefficients, θ_i are the moving average coefficients, e_t is the white noise error term. Since the ARIMA model is well-known in literature, for additional de-

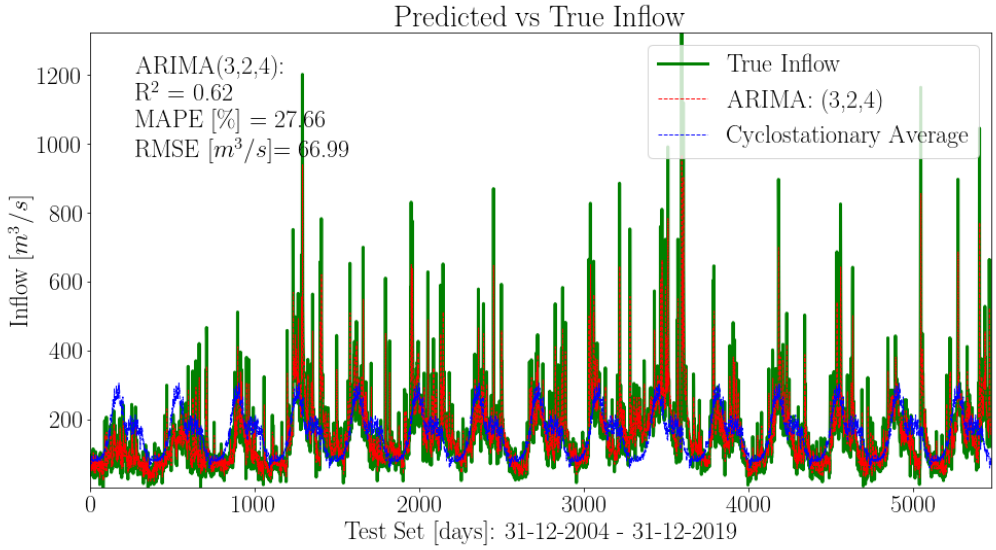

Figure 2: Historical (true) inflow, ARIMA (3,2,4), cyclostationary average: 31-12-2004 - 31-12-2019.

tails please refer to Moayedi and Masnadi-Shirazi (2008), Ariyo et al. (2014).

4. Control Problem

In Equations (2), we show the control problem. This set of equations constitutes the control problem implemented in each control architecture proposed in this work. The control/simulation resolution is daily (e.g, the time-step index h is of 1 day).

$$\min_u \{J_c = c_f||\mathbf{s} - s_{max}||_2 + c_d||\mathbf{s} - s_{min}||_2 + c||\epsilon||_2\} \tag{2a}$$

$$s_{h+1} = s_h + 3600 \cdot 24(\hat{q}_h - u_h) \tag{2b}$$

$$u_{min} \leq u_h \leq u_{max} \tag{2c}$$

$$u_h \geq d_h + \epsilon_h \tag{2d}$$

$$u_h - u_{h-1} \leq 240\ m^3/s \quad if\ l_h \leq 1.1\ m \tag{2e}$$

$$u_h - u_{h-1} \leq 360\ m^3/s \quad if\ l_h > 1.1\ m \tag{2f}$$

$$\mathbf{s} \in R^H, \mathbf{u} \in R^H, \epsilon \in R^H \tag{2g}$$

$$H = 3, c_f = 10^{-4}, c_d = 10^{-3}, c = 100 \tag{2h}$$

Equation (2a) shows the minimization problem. $\mathbf{u}$ $[m^3/s]$ is the control action (e.g., the water release going out from the lake) vector over the prediction horizon H. c_f, c_d and γ are numerical coefficients tuned to have the same order of magnitude for the 3 objectives whose value is collected in Equations (2h), for further details, see Cestari et al. (2022), Cestari et al. (2023). The choice of the prediction horizon $H = 3$ days is based on the unreliability of the forecasting models for longer prediction horizons. $\mathbf{s}$ $[m^3]$ is the lake volume vector. s_{min} and s_{max} are the minimum and maximum lake volumes. The control goal is, simultaneously, maintaining the lake volume between the minimum and maximum thresholds (first and second cost components) and satisfying the agricultural water demand (third component). Equation (2b) shows the daily mass balance equation for the lake volume. $\hat{q}_h$ is the predicted water inflow entering the lake in the h time-step of the prediction horizon. Equation (2c) shows the control action constraints, minimum u_{min} and maximum u_{max} respectively. These constraints are non-linear functions of the lake level. For additional details see Cestari et al. (2022). Equation (2d) shows an additional soft constraint on control action. Water release must be as close as possible to the agricultural water demand d. The constraint must be soft to include cases where the release is lower than the request (dry periods) and where it must be above the request (flood periods). This is ensured through the slack variable ϵ, an additional penalty term in the cost function (2a). Equations (2e) and (2f) ensure that the variation of control action is always below the daily limits imposed by the Olginate dam operation. These boundaries depend on l $[m]$, lake level.

4.1. Controllers

The controllers under study are:

1. **Oracle MPC**: ideal reference controller that sees the actual realization of the inflow on the prediction horizon.
2. **ARIMA-fixed MPC**: a controller that uses the ARIMA(3,2,4) model. Its parameters are identified offline on the historical training dataset (January 1, 1946 - December 31, 2004). The parameters θ are obtained explicitely solving a least-squares problem:

$$\theta = \arg\min\{(q_{hist} - q(\theta))^T \cdot (q_{hist} - q(\theta))\} \tag{3}$$

where $q_{hist} \in R^{T_{hist}}$ is the historical time-series, where $T_{hist} = 21549$ is the length of the training set. $q(\theta)$ is the ARIMA(3,2,4) prediction model described in Equation (1).

3. **ARIMA-recursive MPC**: a controller that uses the ARIMA(3,2,4) model but updates the parameters at every instant via a recursive equation. Details are given in Section 4.2.
4. **BO-MPC**: the controller, contribution of this study, in which the ARIMA is used in prediction and whose parameters are identified, at each time instant, via the iterative BO procedure. Deep insight is given in Section 4.3.

Each controller solves at each time instant, the same optimization problem described in Equations (2). The way in which the prediction model is identified is the only difference between the compared architectures.

4.2. ARIMA-recursive MPC

We introduced this controller in the comparison to provide a benchmark with online tuning capabilities as the BO-MPC. To have a fair comparison, this control strategy involves updating the parameters of the prediction model at each iteration through a moving window (of size W, equal to the backtesting window of the BO-MPC strategy) fashion. In this way, we show that the potential of the BO-MPC does not lie in the online updating of the parameters (considering only the most recent data) but instead in identifying a prediction model that considers both the identification of the history and backtesting performance. More details on the BO-MPC controller are explained in Section 4.3. The hyperparameters characterizing ARIMA-recursive MPC are $W = 5$, set to keep the comparison fair, and $\gamma \in [0.98, 0.995]$ (reference range) forgetting factor of the past data. The strategy is based on the principle of Recursive Infinite-History Estimation. The estimation algorithm minimizes the prediction error on the newly measured data.

$$\theta_t = \theta_{t-1} + \Gamma_t(q_t - \hat{q}_t) \tag{4a}$$

$$\hat{q}_t = \psi_t^T \theta_{t-1} \tag{4b}$$

$$\Gamma_t = Q_t \psi_t \tag{4c}$$

$$Q_t = \frac{P_{t-1}}{\gamma + \psi_t^T P_{t-1} \psi_t} \tag{4d}$$

$$P_t = \frac{1}{\gamma}(P_{t-1} - \frac{P_{t-1}\psi_t\psi_t^T P_{t-1}}{\gamma + \psi_t^T P_{t-1}\psi_t}) > 0 \tag{4e}$$

t is the simulation time index, Γ is the gain, e.g., how much the prediction error $q_t - \hat{q}_t$ affects the update of the parameter estimate. ψ is the gradient of the predicted model output with respect to the parameters θ. The shape of Q depends on the chosen approach (normalized or unnormalized gradient). For deeper insights refer to Carlson (1973), Ljung (1999), and Zhang (2000). P must be a positive-definite matrix. The selected γ are 0.98 and 1. For each of this two choices, the corresponding control performances are shown as part of the controller benchmarking on the regulation of Lake Como basin. Choosing $\gamma = 1$ corresponds to a Kalman filter-based parameter update. Choosing instead $\gamma = 0.98$, a lower forgetting factor, determines a higher penalty on past data. This approach discounts old measurements exponentially such that an observation that is τ samples old carries a weight that is equal to γ^τ times the weight of the most recent observation. $\tau = \frac{1}{1-\gamma}$ represents the *memory horizon* of the algorithm. For further details please refer to Paleologu et al. (2008), Ding et al. (2016), and Islam and Bernstein (2019). The implementation of the algorithm is performed through the Matlab System Identification Toolbox.

4.3. Bayesian-optimized Model Predictive Controller

Equation (5) shows the cost that the BO, at each time step, minimizes to identify the optimal model parameters. This cost includes the identification cost on the historical time-series plus the closed-loop control cost obtained on the backtesting window W, adjusted by a factor λ, hyperparameter that must be tuned. The second term is fundamental to direct the identification of the prediction model for the current time step towards a model that is effective if inserted in the predictive control loop. Our idea is that, thanks to this artificial control-oriented bias we deliberately add to the identification cost, we can obtain a prediction model that allows us to achieve better control performances.

$$J_{BO}(\theta) = J_{ID}(\theta) + \lambda \cdot J_c^W(\theta) \tag{5a}$$

$$J_{ID}(\theta) = (q_{hist} - q(\theta))^T \cdot (q_{hist} - q(\theta)) \tag{5b}$$

Algorithm 1 shows the Bayesian-optimized Model Predictive Controller procedure. At each time step, up to the end of the simulation horizon T, the BO algorithm is executed up to the iteration K. At each BO iteration k, the backtested closed-loop performances are evaluated from time $t - W + w$ up

to t. This allows us to add as regularization term to the identification cost a component that specifies *which could have been the closed-loop performances if the controller would have used the θ_k parameters for the prediction model.*

Algorithm 1 Bayesian-optimized Model Predictive Controller Algorithm

for t = 1,...,T **do**
  for k = 1,...,K **do**
    for w = 1,...,W **do**
      Solve (2), using $\hat{q}(\theta_k)$, at time $t - W + w$.
      Save optimum control cost $J_c^{t-W+w}(\theta_k)$ from Equation (2a).
      Compute the cumulative control cost $J_c^{w+1} = J_c^w + J_c^{t-W+w}(\theta_k)$
    End for.
    Compute $J_{BO}(\theta)$ with Equation (5).
    Sample θ_{k+1} to minimize $J_{BO}(\theta)$ in the next iteration.
  End for.
  Return the optimal θ_t.
  Solve (2), using $\hat{q}(\theta_t)$.
  Return optimal u_t, function of θ_t.
  Execute system dynamics.
  Collect new measured data $q_{hist} = [q_{hist}, q_t]$.
End for.

5. Numerical Simulations and Results

We performed the numerical simulations with real data from 2005 to 2010 (outside the training set). We used the 2005 data to tune the control strategy, leaving 2006 to 2010 for validation. All computations are carried out on an Intel Core i7-8750H with 6 cores, at 2.20 GHz (maximum single core frequency: 4.10 GHz), with 16 GB RAM and running Matlab R2019b. The control system runs in Matlab-Simulink, and the MPC optimization steps are implemented using CVX with the SDPT3 solver (see Grant and Boyd (2008), Grant and Boyd (2014)).

To evaluate the control performances of the different strategies, we compute the control cost *a posteriori*, e.g., the control cost described by Equation (2a) considering the system trajectories (u, s) over the entire simulation horizon. The result is normalized to the cost obtained by the Oracle controller,

which always performs the best. Figure 3 shows the sensitivity analysis concerning the hyperparameter λ to select the optimal trade-off between identification cost and backward closed-loop performances in the BO cost of Equation (5), on year 2005. The y-axis shows the percentage increase of control cost concerning Oracle, defined as $\Delta J_{MPC}[\%] = \frac{J - J_{Oracle}}{J_{Oracle}} \cdot 100$ as function of λ. The x-axis is on logarithmic scale. The best-achieved result is for $\lambda = 10^6$. This shows that the optimal model is achieved by significantly weighting backward closed-loop performances. However, control performances decrease for extremely large values of λ. This highlights the importance of maintaining a relevant contribution from the identification component. Figure 4 shows the trend of identification costs (see Equation

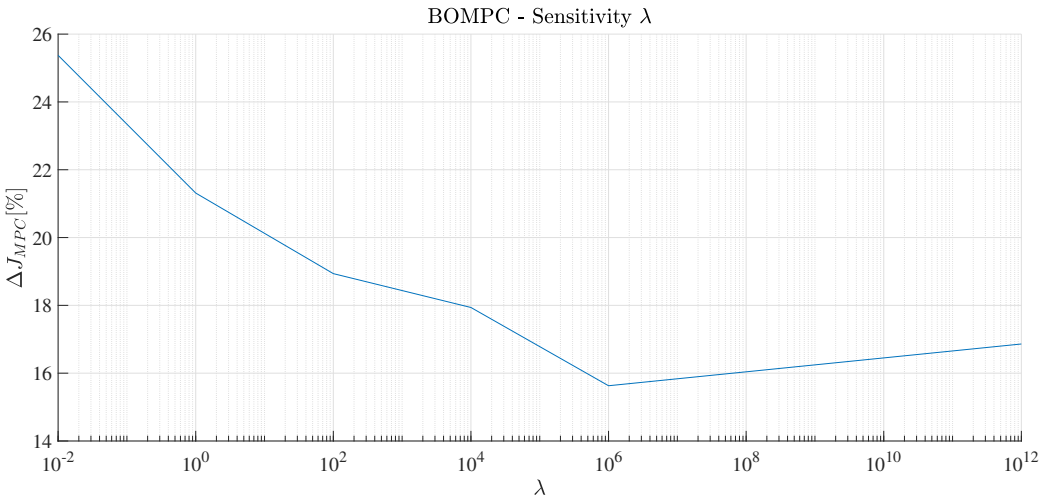

Figure 3: BOMPC - λ sensitivity.

(5b)), instant by instant, over the simulation year 2005, as the parameter λ varies. The reference is the identification cost obtained for the prediction model used by the ARIMA-fixed controller (the constant blue line). As expected, this shows the lowest identification cost. Given the historical data, the corresponding ARIMA(3,2,4) used to predict the inflows is the best prediction model for this class at each instant. Conversely, the identification cost associated with the prediction models used in BO-MPC architectures (for different λ) is always greater (in particular, it grows on average as λ increases, e.g., a larger closed-loop control bias is introduced). Nonetheless, as the following dissertation shows, in a control perspective the best prediction model is not the one with the lowest identification cost. Figure 5 shows the

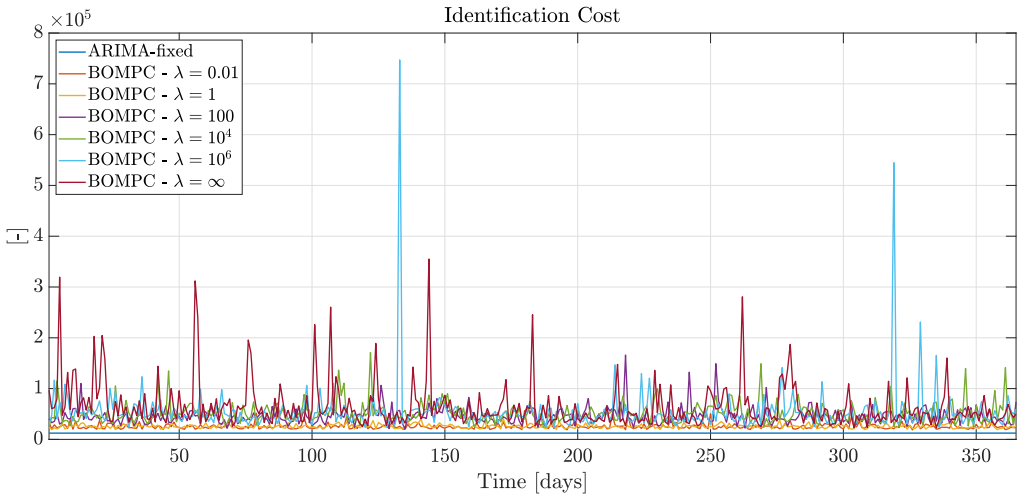

Figure 4: Identification Cost BO-MPC - λ sensitivity.

control cost increment concerning Oracle controller for the control architectures described in Section 4.1. The ARIMA controllers (green) show higher control cost than the best BO-MPC strategy ($\lambda = 10^6$). The ARIMA-fixed controller is the best among the remaining controllers. This indicates that the strength of the BO-MPC algorithm does not lie in the online update of the parameters but in the presence of the closed-loop control term in the prediction model identification. By choosing $\lambda = 0.01$, we observe that the control performance of BO-MPC is worse than ARIMA-fixed. This depends on the fact that the regularization term linked to the closed-loop performance needs to be weighted more: its effect is introducing a misleading bias without improving performances. Figure 6 shows the system trajectories over the 2007. Oracle, ARIMA-fixed, BO-MPC, and historical regulation control strategies are shown. The upper panel shows the lake level. The panel in the middle shows the water release. The lower panel shows the water inflow. The historical regulation determines less efficient trajectories: water release is often lower than water demand, and in the period between the 160th and 180th day, there is evidence of higher water release than the downstream demand. This leads to a higher deficit in the following period. Additionally, it fails to prevent the lake level from going below the lower limit. This behavior is shown in the majority of the tested years and motivates the application of the automatic control strategy offered by the MPC. Figure 6 shows that BO-MPC makes the system stick to lower levels, maximizing the downstream

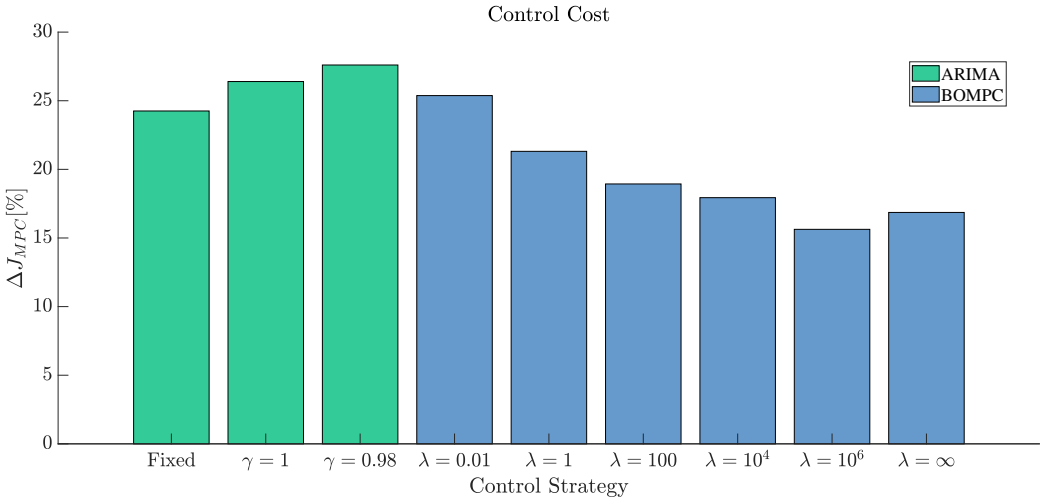

Figure 5: Control Cost - ARIMA MPCs vs BO-MPC for varying λ - 2005.

water release, and satisfying water demand, even in dry periods. This is the significant difference concerning ARIMA-fixed MPC. Indeed, it depends on the ARIMA(3,2,4) prediction, fitted on historical time-series. Whenever the forecast fails, the corresponding control action is consequently suboptimal. This leads to a system trajectory far from the lower bounds and determines lower water release and, thus, a higher deficit. This is particularly evident between the 150th and 200th days, as the black arrows indicate. BO-MPC matches the Oracle system trajectory, thanks to the closed-loop term in J_{BO}, whereas ARIMA-fixed trajectory is further from the Oracle. The oscillations in the control action present in both the ARIMA-fixed and BO-MPC configurations do not prevent the satisfaction of the physical constraints of the dam on the control variation set at 240 m^3/s per day in regular operation and 360 m^3/s in flooded condition (level greater than or equal to 1.1 m), as specified in the MPC constraints of Equations (2e) and (2f). Figure 7 and Table 3 show ΔJ_{MPC} [%] and average inflow from 2005 to 2010. We observe that:

1. BO-MPC always shows lower ΔJ_{MPC} [%].
2. BO-MPC is significantly improving the performances, making the strategy closer to the Oracle benchmark in dry periods characterized by lower average inflow.
3. In wet years, the advantage of BO-MPC strategy is less effective (per-

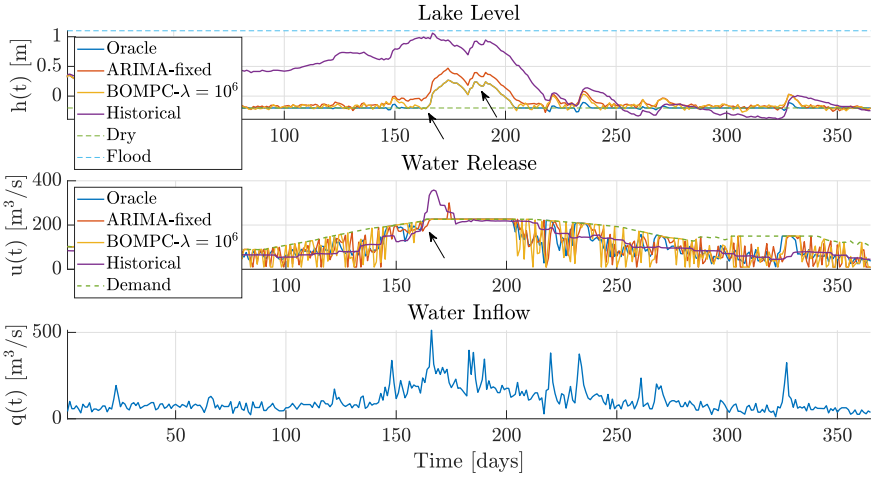

Figure 6: Oracle, ARIMA-fixed, BO-MPC, Historical - 2007.

formances are closer to ARIMA-fixed ones). However, both ARIMA-fixed and BO-MPC performances are close to the Oracle one (ΔJ_{MPC} is almost 0 %). This indicates that, in wet years, given this control framework, the quality of the prediction model has a negligible effect on the quality of the control performances. This conclusion is intuitive if considering the physics of the system. In wet regimes, the lake volume is high; thus, the water inflow impact on the overall water volume is minor. The controller computes the optimal release giving priority to the lake volume management. Figure 7 is a graphical representation of Table 3. The size of the markers is proportional to the average inflow in the same year.

Table 3: $\Delta J_{MPC}[\%]$ - Validation

	ARIMA-fixed	**BO-MPC**	**Avg. Inflow** $[m^3/s]$
2005	24.2	15.6	86.1
2006	19.3	17.9	108.8
2007	19.3	2.51	107.2
2008	1.90	1.80	192.0
2009	0.54	0.25	160.5
2010	1.70	0.61	184.2

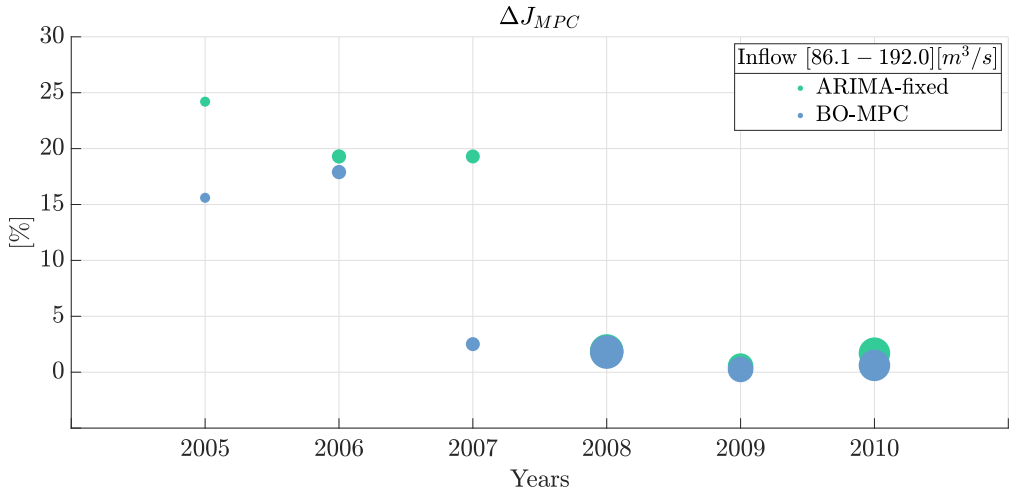

Figure 7: ΔJ_{MPC} [%], ARIMA-fixed and BO-MPC, 2005-2010.

Figure 8 shows the elapsed time, at each time step, for the execution of the BO-MPC algorithm. The average time is below 2 minutes. Considering system dynamics time resolution of 1 day, the strategy is affordable in a practical deployment.

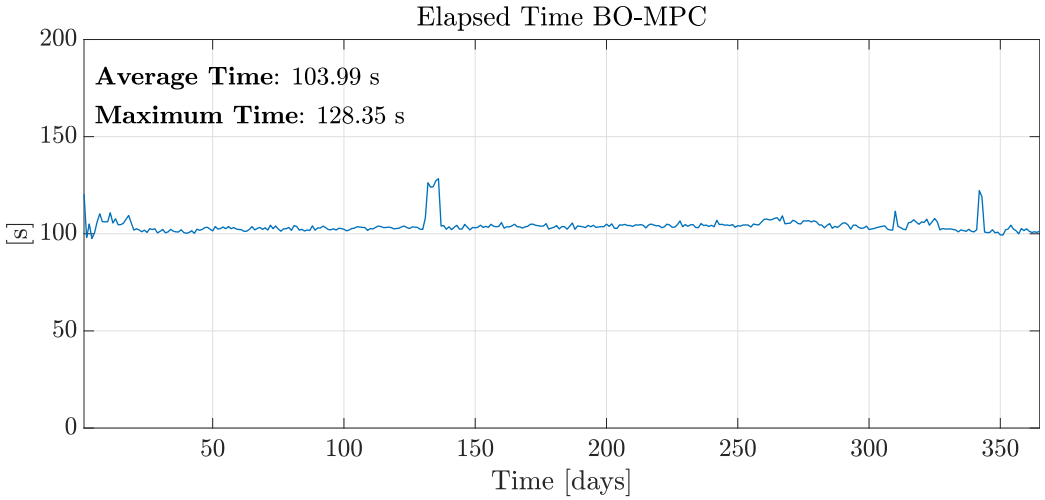

Figure 8: Elapsed time at each iteration of the BO-MPC algorithm, over the simulation horizon.

6. Conclusions and Future Work

In this work, we show the impact of introducing a regularization term related to closed-loop performance when identifying a prediction model in predictive control. Thanks to the Bayesian Optimization algorithm, we can find, at each time-step, the best prediction model belonging to the ARIMA(3,2,4) class that maximizes the performance of the closed-loop predictive controller. This approach proves to be particularly effective in managing dry periods, as the quality of the prediction model improves the management of scarce water resources satisfying the agricultural district. The computational load is sustainable as one simulation step requires on average less than 2 minutes, against the system dynamics of 1 day. Therefore, the algorithm is directly applicable to the actual system. The approach was validated and tested on 6 years of real data to calibrate the hyperparameters and evaluate the strategy. The BO-MPC algorithm always performs better than the ARIMA-fixed benchmark, which exploits the available data only to fit historical data. In future work, we plan to conduct an extensive validation campaign to demonstrate the methodology's effectiveness on other hydrogeological basins. We will also test additional technologies for online hyperparameter calibration to enrich the comparison and establish, within the prediction for control framework, which are the best.

References

Abbracciavento, F., Zinnari, F., Formentin, S., Bianchessi, A.G., Savaresi, S.M., 2023. Multi-intersection traffic signal control: A decentralized mpc-based approach. IFAC Journal of Systems and Control 23, 100214.

Ariyo, A.A., Adewumi, A.O., Ayo, C.K., 2014. Stock price prediction using the arima model, in: 2014 UKSim-AMSS 16th international conference on computer modelling and simulation, IEEE. pp. 106–112.

Carlson, N.A., 1973. Fast triangular formulation of the square root filter. AIAA journal 11, 1259–1265.

Cestari, R.G., Castelletti, A., Formentin, S., 2022. Hourly operation of a regulated lake via model predictive control. IFAC-PapersOnLine 55, 7–12.

Cestari, R.G., Castelletti, A., Formentin, S., 2023. Scenario-based model predictive control of water reservoir systems. https://doi.org/10.48550/arXiv.2309.00373 .

Denaro, S., Anghileri, D., Giuliani, M., Castelletti, A., 2017. Informing the operations of water reservoirs over multiple temporal scales by direct use of hydro-meteorological data. Advances in water resources 103, 51–63.

Dewancker, I., McCourt, M., Clark, S., 2016. Bayesian optimization for machine learning: A practical guidebook. arXiv preprint arXiv:1612.04858 .

Ding, F., Wang, X., Chen, Q., Xiao, Y., 2016. Recursive least squares parameter estimation for a class of output nonlinear systems based on the model decomposition. Circuits, Systems, and Signal Processing 35, 3323–3338.

Elsaraiti, M., Merabet, A., 2021. A comparative analysis of the arima and lstm predictive models and their effectiveness for predicting wind speed. Energies 14, 6782.

Faber, B.A., Stedinger, J., 2001. Reservoir optimization using sampling sdp with ensemble streamflow prediction (esp) forecasts. Journal of Hydrology 249, 113–133.

Ficchì, A., Raso, L., Dorchies, D., Pianosi, F., Malaterre, P.O., Van Overloop, P.J., Jay-Allemand, M., 2016. Optimal operation of the multireservoir system in the seine river basin using deterministic and ensemble forecasts. Journal of Water Resources Planning and Management 142, 05015005.

Grant, M., Boyd, S., 2008. Graph implementations for nonsmooth convex programs, in: Blondel, V., Boyd, S., Kimura, H. (Eds.), Recent Advances in Learning and Control. Springer-Verlag Limited. Lecture Notes in Control and Information Sciences, pp. 95–110.

Grant, M., Boyd, S., 2014. CVX: Matlab software for disciplined convex programming, version 2.1.

Hutter, F., Kotthoff, L., Vanschoren, J., 2019. Automated machine learning: methods, systems, challenges. Springer Nature.

Islam, S.A.U., Bernstein, D.S., 2019. Recursive least squares for real-time implementation [lecture notes]. IEEE Control Systems Magazine 39, 82–85.

Le, V.A., Malikopoulos, A.A., 2023. Optimal weight adaptation of model predictive control for connected and automated vehicles in mixed traffic with bayesian optimization, in: 2023 American Control Conference (ACC), IEEE. pp. 1183–1188.

Lévesque, J.C., Gagné, C., Sabourin, R., 2016. Bayesian hyperparameter optimization for ensemble learning. arXiv preprint arXiv:1605.06394 .

Ljung, L., 1999. System identification-theory for the user 2nd edition ptr prentice-hall. Upper Saddle River, NJ .

Lucchini, A., Formentin, S., Corno, M., Piga, D., Savaresi, S.M., 2020. Torque vectoring for high-performance electric vehicles: an efficient mpc calibration. IEEE Control Systems Letters 4, 725–730.

Moayedi, H.Z., Masnadi-Shirazi, M., 2008. Arima model for network traffic prediction and anomaly detection, in: 2008 international symposium on information technology, IEEE. pp. 1–6.

Mosavi, A., Ozturk, P., Chau, K.w., 2018. Flood prediction using machine learning models: Literature review. Water 10, 1536.

Paleologu, C., Benesty, J., Ciochina, S., 2008. A robust variable forgetting factor recursive least-squares algorithm for system identification. IEEE Signal Processing Letters 15, 597–600.

Piga, D., Forgione, M., Formentin, S., Bemporad, A., 2019. Performance-oriented model learning for data-driven mpc design. IEEE Control Systems Letters 3, 577–582. doi:10.1109/LCSYS.2019.2913347.

Sagheer, A., Kotb, M., 2019. Time series forecasting of petroleum production using deep lstm recurrent networks. Neurocomputing 323, 203–213.

Sangiorgio, M., Dercole, F., 2020. Robustness of lstm neural networks for multi-step forecasting of chaotic time series. Chaos, Solitons Fractals 139, 110045. URL: https://www.sciencedirect.com/science/article/pii/S0960077920304422, doi:https://doi.org/10.1016/j.chaos.2020.110045.

Sangiorgio, M., Dercole, F., Guariso, G., 2021. Forecasting of noisy chaotic systems with deep neural networks. Chaos, Solitons & Fractals 153, 111570. URL: https://www.sciencedirect.com/science/article/pii/S0960077921009243, doi:https://doi.org/10.1016/j.chaos.2021.111570.

Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., De Freitas, N., 2015. Taking the human out of the loop: A review of bayesian optimization. Proceedings of the IEEE 104, 148–175.

Sorourifar, F., Choksi, N., Paulson, J.A., 2023. Computationally efficient integrated design and predictive control of flexible energy systems using multi-fidelity simulation-based bayesian optimization. Optimal Control Applications and Methods 44, 549–576.

Stenger, D., Ay, M., Abel, D., 2020. Robust parametrization of a model predictive controller for a cnc machining center using bayesian optimization. IFAC-PapersOnLine 53, 10388–10394.

Victoria, A.H., Maragatham, G., 2021. Automatic tuning of hyperparameters using bayesian optimization. Evolving Systems 12, 217–223.

Wu, J., Chen, X.Y., Zhang, H., Xiong, L.D., Lei, H., Deng, S.H., 2019. Hyperparameter optimization for machine learning models based on bayesian optimization. Journal of Electronic Science and Technology 17, 26–40.

Zhang, Q., 2000. Some implementation aspects of sliding window least squares algorithms. IFAC Proceedings Volumes 33, 763–768.