Systems & Control Transactions

Research Article - Peer Reviewed Conference Proceeding
ESCAPE 35 - European Symposium on Computer Aided Process Engineering
Ghent, Belgium. 6-9 July 2025
Jan F.M. Van Impe, Grégoire Léonard, Satyajeet S. Bhonsale,
Monika E. Polańska, Filip Logist (Eds.)

PSE PRESS

Application of pqEDMD to Modeling and Control of Bioprocesses

Camilo Garcia-Tenorio[a]*, Guilherme A. Pimentel[a], Laurent Dewasme[a], and Alain Vande Wouwer[a]

[a] Systems, Estimation, Control and Optimization (SECO), University of Mons, 7001, Mons, Belgium
* Corresponding Author: camilo.garciatenorio@umons.ac.be

ABSTRACT

Extended Dynamic Mode Decomposition (EDMD) and its variant, the pqEDMD, which uses a p-q-quasi norm reduction of polynomial basis functions, are attractive tools to derive linear operators approximating the dynamic behavior of nonlinear systems. This study highlights how this methodology can be applied to data-driven modeling and control of bioprocesses by discussing the selection of several ingredients of the method, such as the polynomial basis, order, data sampling, and preparation for training and testing, and ultimately, the exploitation of the model in linear model predictive control.

Keywords: System Identification, Dynamic Modelling, Model Predictive Control, Process Control, Numerical Methods

INTRODUCTION

Digital twins are increasingly popular as part of the Industry 4.0 revolution but are generally understood as digital representations of the industrial processes obtained through machine learning techniques and, in particular, the use of neural networks (NN). Although NNs are versatile and have proven their utility in many applications, there exist less popular yet appealing alternatives offered by the stream of research that revolves around the Koopman operator approximation [1–3], and the different decomposition methods to get an approximation of the Koopman operator [4]. In particular, the Extended Dynamic Mode Decomposition (EDMD) provides the possibility to derive a linear representation of a nonlinear system in a so-called “function space” of nonlinear basis functions (denoted “observables”) [5].

The objective of this study is to show the potential of EDMD [5,6] and one of its extensions, pqEDMD, developed by the present authors in [7] for modeling arbitrary systems, including bioprocesses. The main advantage of providing a linear representation is that standard linear estimation and control techniques can be further exploited, particularly linear model predictive control (MPC), whereas NNs require more complex nonlinear methods [8]. The prominent feature of pqEDMD is that it systematizes the definition of the dictionary of observable functions by resorting to orthogonal polynomials and selecting the approximation order by sweeping through two parameters, denoted p and q.

This paper explains how the approximation can be constructed, discusses practical aspects such as the selection of training and testing data, and shows how the resulting models can be exploited in linear MPC as applied to a simple yet representative bioreactor case study.

MODELING AND CONTROL USING PQEDMD

This study considers a data-driven modeling strategy called the p-q-quasi-norm extended dynamic mode decomposition (pqEDMD) that takes its origins in the EDMD formulation from Williams, et al., [5]. This method applies a p-q-quasi norm reduction to the set of observables, limiting the maximum order of the orthogonal polynomials that compose the basis for the approximation. Following the approach for incorporating the system inputs in the EDMD decomposition [3], the overall approach is to approximate the original nonlinear dynamics with a linear operator and to apply linear control methods.

pqEDMD Model Derivation

Consider an autonomous nonlinear system in

discrete-time, with state variables $x(k) \in \mathbb{R}^n$, discrete-time $k \in \mathbb{Z}_0^+$, and forcing signal $u(k) \in \mathbb{R}^m$, described by the following difference equation:

$$x(k+1) = T\big(x(k)\big) + Bu(k) \tag{1}$$

where $T: \mathbb{R}^n \to \mathbb{R}^n$ is the differentiable vector-valued evolution map, i.e., the nonlinear state transition rule, and $B \in \mathbb{R}^{n\times m}$ is the input matrix, defining the system as input affine. Note that the output of the system is the full state. An orbit of the system is the sequence of states $\{x_i\}_{i=0}^{k}$ that comes from the solution of (1). The sequence is the successive application of the non-linear mapping T from an initial condition $x_0 \in \mathbb{R}^n$ at $k = 0$ and a specific sequence of forcing signals $u \triangleq \{u_i\}_{i=0}^{k-1}$. It is possible to group each orbit in a set of tuples $\{(x_i, y_i, u_i)\}$, where $y_i = T(x_i) + Bu_i$. Then, organize these tuples into the data matrices,

$$X = [x_1 \quad \dots \quad x_N],\ Y = [y_1 \quad \dots \quad y_N],\ U = [u_1 \quad \dots \quad u_N]. \tag{2}$$

The rationale behind the approach is to get linear predictors of a function space of observables $\Psi(x) = [\psi_1(x), \cdots, \psi_d(x)]^\top: \mathbb{R}^n \to \mathbb{C}^d$, where the elements of the function space come from a family of orthogonal polynomials, and the use of q-quasi norms with a maximum order p to reduce the maximum order and quantity of elements in the observables [7]. The condition that these observables must satisfy is:

$$\Psi(y) = A\Psi(x) + Bu + r(x,u) \tag{3}$$

where $r(x,u) \in \mathcal{F}$ is the residual term to minimize in order to find matrices A and B. This leads to the least squares problem,

$$\|r(x,u)\|^2 = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\|\Psi(y_i) - A\Psi(x_i) - Bu\|_2^2.$$

The evaluation of the data matrices (2) with the set of observables produces the observed data matrices $\Psi(\{X,Y\}) \in \mathrm{R}^{d\times N}$. Then, from the singular value decomposition of the observed state $\Psi(X)^T = \mathcal{U}\mathcal{S}\mathcal{V}^T$, it is possible to determine the effective rank of $\Psi(X)$ using the singular values $\sigma_1, \sigma_2, \cdots, \sigma_d = \mathrm{diag}(\mathcal{S})$. The effective rank or ϵ-rank $r_\epsilon = \min\{r: \sigma_r \le \epsilon N \sigma_1\}$, where $\epsilon << 1$, an arbitrarily small number, e.g., eps in Matlab.

From the effective rank, $\mathcal{U}_r = \mathcal{U}_{:,1:r_\epsilon}$, $\mathcal{S}_r = \mathcal{S}_{1:r_\epsilon,1:r_\epsilon}$ and $\mathcal{V}_r = \mathcal{V}_{:,1:r_\epsilon}$, and the solution becomes:

$$\begin{aligned} D_r &= \mathcal{S}_r \backslash \mathcal{U}_r^T \Psi(Y)^T \\ [A \quad B]^T &= \mathcal{V}_r D_r. \end{aligned} \tag{4}$$

Notice that the solution for the transition matrix A, i.e., equation (4), cannot guarantee the convergence of the function space into a stable attractor, i.e., A may not be Hurwitz (all the magnitudes of its eigenvalues less or equal to one). Then, some solutions for some p-q parametrization give trajectories of observables that diverge to infinity. This guarantee is an open problem and highlights the importance of testing many combinations of p-q parameters on a particular dataset.

As the purpose of linear predictors, as in (3), is the synthesis of controllers, it is necessary to recover the state from the observables. Usually, the recovery of the state requires the solution of an additional least squares problem. Instead, we exploit the structure of the order-one polynomials in the set of observables and recover the state analytically. The order-one polynomials are linear functions of the individual states. Therefore, the C matrix comes from converting the inverse of the order-one polynomials into a matrix form.

$$C = \Big[c_p \quad ; \quad \mathrm{diag}\{\psi_i^{-1}(x)\}_{i=1}^{n}; \quad 0_{(n,d-n-1)}\Big] \tag{5}$$

where $c_p \in \mathbb{R}^n$ is a vector where every element is the constant term of the order-one polynomials. This vector is concatenated with a diagonal matrix of the transformation into matrix form of the function inverse of the non-constant term of the order-one observables. The remaining entries for the higher-order polynomials are zero. The result is a linear operator A acting on a function space Ψ and an input matrix B. The linear time-evolution of the functions $\psi(x) \in \Psi(x)$ is related to the nonlinear time-evolution of the state x. Matrix C achieves the transformation back to the original state space. Hence, the original nonlinear system (1) becomes:

$$\begin{aligned} \Psi\big(x(k+1)\big) &= A\Psi\big(x(k)\big) + Bu(k) \\ x(k) &= C\Psi\big(x(k)\big), \end{aligned} \tag{6}$$

pqEDMD for MPC

For driving the system into a desired state, consider a model predictive control algorithm adapted from [9], and an MPC formulation in a function space similar to the developments by Korda et al. [3]. The objective is to find a sequence of inputs u_k starting at the k^{th} output measurement time-instant that spans a control horizon N_c, the sequence of inputs, drives the system to a desired output by minimizing a cost function over a prediction horizon N_p while satisfying some input/output inequality constraints. The MPC formulation over the function space is

$$\begin{aligned} \min_u J(x,u) = &\sum_{i=0}^{N_p-1}\Big(\Psi_{\text{dev}}\big(x(i)\big)^T Q_\psi \Psi_{\text{dev}}\big(x(i)\big)\Big) \\ &+ \sum_{i=0}^{N_c-1}\big(u(i)^T Q_u u(i)\big) \\ &+ \Psi_{\text{dev}}\Big(x(N_p)\Big)^T F_\psi \Psi_{\text{dev}}\Big(x(N_p)\Big) \end{aligned} \tag{7}$$

subject to

$$u_{\min} \le u^i \le u_{\max}$$

where $\Psi_{\text{dev}}\big(x(i)\big) = \Psi\big(x(i)\big) - \Psi\big(x_{\text{ref}}(i)\big)$, the difference of the function space with its reference. The symmetric matrices $Q_\psi, F_\psi \in \mathbb{R}^{d\times d}$ are the weight matrices for the state sequence, and the terminal state and the symmetric matrix $Q_u \in \mathbb{R}^{m\times m}$ is the weighting matrix for the control inputs. The weighting matrix F_ψ of the terminal state accounts for different prediction and control horizons and imposes a suitable terminal constraint to ensure stability. In contrast with the traditional MPC formulation, equation (7) is an objective function in the function space and

not the state space of a traditional linear system. It is necessary to transform the formulation (7) into a quadratic program in the function space, i.e., a quadratic objective function $\Delta u^T H \Delta u + h^T \Delta u$ with linear constraints $L\Delta u < b$, that has a solution using a standard solver.

The first step is to derive the observable weighting matrices Q_ψ and F_ψ from selected output weighting matrices Q_y and F_y. The first method is to exploit the output matrix C from the pqEDMD solution, i.e.,

$$Q_\psi = C^T Q_y C, \tag{8}$$

This formulation restricts the MPC optimization to the convergence of the order-one polynomials to their reference values, neglecting the fact that the remaining elements in the observables also depend on the outputs. Even if that is the case, formulating Q_ψ this way gives good results and fast convergence. The second method uses the observables to transform the matrix,

$$Q_\psi = \text{diag}\left(\Psi\left(\text{diag}(Q_y)\right)\right) \tag{9}$$

This method weighs all the observables in the basis and should produce more accurate solutions. Performance-wise, the convergence of this solution is slower, and it may break the feasibility of the quadratic problem. These methods also apply to calculating F_ψ.

Continuing with the translation of the problem into a quadratic program, consider the solution of the identified dynamics in the function space $\Psi(x)$, a solution of (6) from an initial condition $x(0)$, and a sequence of inputs $\{u_i\}_{i=0}^{k-1}$ up to an arbitrary time k,

$$\Psi\big(x(k)\big) = A^k \Psi\big(x(0)\big) + \sum_{j=0}^{k-1} A^{k-1-j} B u(j), \tag{10}$$

Stacking up the state and the input sequences in column vectors and applying the solution of the pqEDMD (10) from a time-step k and up to a prediction horizon N_p gives

$$\underbrace{\begin{bmatrix} \Psi\big(x(k+1)\big) \\ \Psi\big(x(k+2)\big) \\ \vdots \\ \Psi\left(x(k+N_p)\right) \end{bmatrix}}_{\Psi_X} = \hat{A}\Psi\big(x(k)\big) + \hat{B} \underbrace{\begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+N_c) \end{bmatrix}}_{\hat{U}} \tag{11}$$

where $\hat{A}$ and $\hat{B}$ are the block matrices that relate the initial evaluation of the observables $\Psi\big(x(k)\big)$ and the sequence of inputs U to the sequence of observables Ψ_X,

$$\hat{A} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{N_p} \end{bmatrix}, \; \hat{B} = \begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N_p-1}B & A^{N_p-2}B & \cdots & B \end{bmatrix}, \tag{12}$$

where the number of block columns in $\hat{B}$ depends on the control horizon. In addition to the compact form of the sequence of observables (11), the compact form of the objective function (7) is,

$$J(x,u) = \; (\Psi_X - \Psi_{\text{ref}})^T Q_\Psi (\Psi_X - \Psi_{\text{ref}}) + U^T Q_U U, \tag{13}$$

where Q_Ψ is a block diagonal matrix with the observable matrix repeated N_p times along the diagonal and with F_ψ as the final element, and Q_U is a block diagonal matrix with the weighting matrix for the control input repeated N_c times along the diagonal,

$$\begin{aligned} Q_\Psi &= \; \text{diag}\left(\{Q_\psi\}_{i=1}^{N_p}, F_\psi\right), \\ Q_U &= \; \text{diag}\left(\{Q_u\}_{i=1}^{N_c}\right). \end{aligned} \tag{14}$$

With the definition of the sequence of observables and the objective function in compact form, replacing solution (11) into (13) and expanding produce a quadratic element and a linear element in U, among other terms that do not depend on U, i.e.,

$$\begin{aligned} J(x,u) = \; & U^T \hat{B}^T Q_X \hat{B} U + U^T Q_U U \\ & + 2\big(\hat{A}\Psi\big(x(k)\big) - \Psi_{\text{ref}}\big)^T Q_\Psi \hat{B} U + r(x) \end{aligned} \tag{15}$$

where $r(x)$ are the terms that do not depend on U. Then, from (15) the Hessian is

$$H = \hat{B}^T Q_\Psi \hat{B} + Q_U, \tag{16}$$

and from the linear term of the compact cost function (15), the linear term of the quadratic program becomes

$$h^T = 2\big(\hat{A}\Psi\big(x(k)\big) - \Psi_{\text{ref}}\big)^T Q_\Psi \hat{B}, \tag{17}$$

that satisfies the objective of having the problem as a quadratic program, meaning that it is possible to calculate the optimal sequence of inputs via a standard solver, e.g., quadprog() in Matlab.

BIOPROCESS APPLICATION

Consider a continuously stirred tank reactor (CSTR), where an arbitrary biological species grows on a substrate.

Assuming that both substrate and biomass concentrations are measurable, it is possible to get a description of their dynamics by applying a mass balance, resulting in the following differential equations:

$$\begin{aligned} \dot{x_1} &= x_1(\mu(x_2) - D), \\ \dot{x_2} &= D\big(x_{2,f} - x_2\big) - \frac{\mu(x_2) x_1}{\eta}, \end{aligned} \tag{18}$$

where x_1 is the first state, representing the biomass concentration x_2 is the second state, representing the substrate concentration, and $x_{2,f}$ is substrate inlet concentration, and a Haldane kinetics law represents the growth rate:

$$\mu(x_2) = \frac{\mu_{\max} x_2}{k_m + x_2 + k_1 x_2^2}, \tag{19}$$

where the parameter $\mu_{\max}$ is the maximum specific growth rate, k_m is the activation constant, and k_1 is the inhibition constant. Finally, η is the process yield and $D = F/V$ $[h^{-1}]$ the dilution rate. The following assumes that the bioreactor operates in chemostat mode (continuous feeding with constant volume). Table 1 reports the

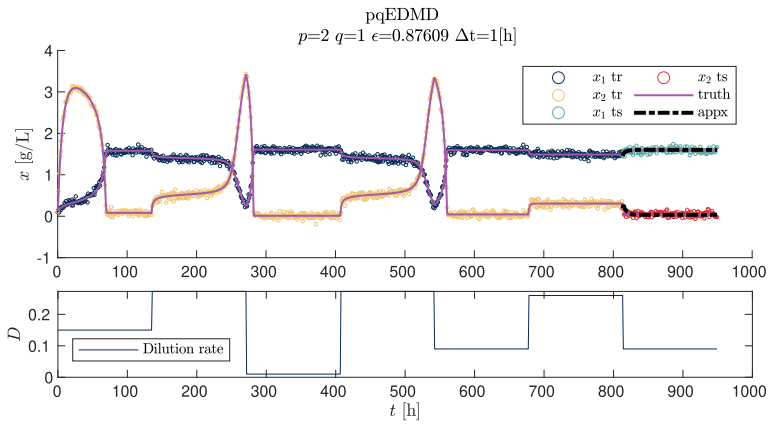

Figure 1: pqEDMD identification with fast sampling and high level of noise.

process parameter values.

Table 1: Reactor Parameters.

Symbol	Value	Units
μ_{max}	0.4	[h^{-1}]
k_m	0.12	[g/L]
k_1	0.4545	[g/L]$^{-1}$
η	0.4	[-]

NUMERICAL RESULTS

Model Training and Testing

To have a rich enough dataset for identification, it is necessary to select the time evolution of the dilution rate and the duration of the experiment. Two additional parameters determine the algorithm's performance: the sampling rate and the standard deviation of the measurement noise. For the time evolution of the dilution rate, the choice is a sequence of inputs that switch the convergence of the trajectories between the two asymptotically stable points. For the sampling and noise, we will consider three different scenarios that cover different values of the parameters: a scenario with fast sampling and low levels of noise, a second case, where we increase the uncertainty in the measurement, and a third, where we decrease the sampling rate. Table 2 shows the exact values of these parameters in the description of each scenario. For illustration, Figure 1 shows the second case, where there is fast sampling and high levels of noise.

In each scenario, the data set is made of a single experiment that starts from low values of biomass and substrate and is subject to 7 different steps of dilution rate, where each step has the same duration. With fast sampling, the data sets contain 951 points. The first 6 steps correspond to the training (814 points), while the rest (137 points) constitute the testing set. With slow sampling, there are 191 points in total, 162 for training, and 29 for testing.

It is possible to apply pqEDMD with a variety of orthogonal polynomials. In this application, their choice is not critical, and the following algorithm implementation uses Legendre polynomials. For some other applications, the choice of polynomials may be important. Also, for applications where the order of magnitude of the different measurements is not the same, normalizing the data may improve the numerical conditioning of the methods. This aspect is also not influential in the present application.

Each element of the observables comes from the product of univariate polynomials with a particular order α_j for each state variable x_j being *observed /evaluated*. Therefore, it is possible to define each observable via a nonnegative vector $\alpha \in \mathbb{N}_+^n$ containing the set of integers corresponding to the orders of the univariate polynomials. A particular element of the set of observables is

$$\psi = \prod_{j=1}^{n} \pi^{\alpha_j}(x_j), \tag{20}$$

where $\pi^{\alpha_j}(x_j)$ is a univariate element from a family of orthogonal polynomials of order α_j. It is possible to apply a selection criterion to each vector of integers α based on a q-quasi norm, which imposes a maximum order p,

$$\alpha = \{\alpha \in \mathbb{N}_+^n : \|\alpha\|_q \leq p\}, \tag{21}$$

where $\|\alpha\|$ is the q-quasi norm of the set of orders, and $p \in \mathbb{N}$ is a positive integer that determines the maximum order of the multivariate polynomial function ψ. The definition of a q-quasi norm of a vector α is

$$\|\alpha\|_q = \left(\sum_{i=1}^{n} \alpha_i^q\right)^{\frac{1}{q}}, \tag{22}$$

where $q \in \mathbb{R}$ defines the quasi norm, and the cardinality of the vector is n because each element represents the order of a univariate polynomial for each of the n states in the original system. In general, it is not a norm because q can take noninteger values (for those cases, equation (22) does not satisfy the triangle inequality).

Table 2: Empirical error p-q parameters and basis size for the best performing and worst performing sets of observables from the p-q sweep.

	Scenario											
	(1) $\Delta t = 1$ [h], $\sigma = .005$ [g/L]^2				(2) $\Delta t = 1$ [h], $\sigma = .05$ [g/L]^2				(3) $\Delta t = 4$ [h], $\sigma = .05$ [g/L]^2			
p	4	5	5	4	2	2	5	5	2	2	3	3
q	2.5	1.5	0.5	2.0	1.0	0.5	0.5	1.5	0.5	1.0	0.5	1.0
d	18	24	12	17	6	5	12	24	5	6	7	10
ϵ	0.17	0.18	0.21	0.22	0.84	0.87	1.12	1.14	0.91	0.96	1.32	1.33

By applying a p-q norm reduction to the polynomial orders, we can significantly reduce the number of observables and their maximum order. This, in turn, enhances the algorithm's numerical accuracy and reduces computational time.

The answer to the question of the optimal p-q parameters to use for a particular dataset is still open. Large values of p-q pairs produce higher order polynomials, which increase the computational complexity of the algorithm without a guarantee of a more accurate approximation. The same effect applies to the uncertainties in the measurements, high-order polynomials tend to amplify the noise leading to unfeasible solutions. Therefore, the method to find the best combination is to perform a sweep over a set of candidate values. For each scenario, the candidate values are $p = [2\ 3\ 4\ 5]$ and $q = [0.5\ 1\ 1.5\ 2\ 2.5]$, giving 13 sets of unique observables to test (some combinations of p-q parameters give redundant sets of observables). Table 2 shows a summary of the results for the three scenarios, where ϵ is the empirical error of the test set that comes from the solution of (6) from the initial conditions of the test set and the application of the last step of the dilution rate. The definition of the empirical error is

$$\epsilon = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{y}_i - y_i|}{|y_i|}, \tag{23}$$

where $\hat{y}$ is the approximation of the output and y is the output from the dataset.

The results from Table 2 show that for the same underlying dynamics, there is no unique pair of p-q parameters that fit the data. In a deterministic case (no noise), with a very fast sampling time $\Delta t = 0.001$ [h] the *optimal* p-q pair is the same as the best approximation for the scenario with fast sampling and low levels of noise, but it is different from the other two scenarios. If the experimental setup involves a slow sampling, like the third scenario, the best p-q pair for the deterministic case produces a pqEDMD solution that diverges, highlighting the importance of a p-q sweep.

The experiments vary the dilution rate between low and high values. The high value of the dilution rate $D = 0.273$ gives the wash-out point a big region of attraction. Then, the value of the biomass starts to slowly decay until a change in the dilution rate brings the state back to the desired attractor (i.e., operating point). These changes in the concentrations of biomass and substrate give the pqEDMD the necessary information to characterize the system. Otherwise, without this dynamic spectrum, it would be necessary to perform more experiments with varying initial conditions.

In conclusion, it is possible to identify the dynamics of the nonlinear CSTR reactor via the pqEDMD algorithm when there is enough information on the system dynamics in an experimental dataset. The choice of orthogonal polynomials under a p-q-quasi norm reduction makes the algorithm versatile enough to identify the dynamics accurately. The advantage of having a linear representation of a dynamical system in a function space is the interpretability of the system, from the spectrum of the A matrix to the analysis via theoretical concepts like the Koopman operator (restricted to the case where the pqEDMD converges to the operator).

Linear Model Predictive Control

Figure 2 shows the application of the MPC control adapted to work in a function space instead of a state space for the three scenarios. Table 3 lists the controller parameters, which are the same across all scenarios.

Table 3: MPC parameters.

Symbol	Value	Units
N_p	10	[-]
N_c	7	[-]
Q_y	$\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}$	[-]
Q_u	0.5	[-]
F_y	$\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}$	[-]
$u_{\min}$	0.01	[h^{-1}]
$u_{\max}$	2	[h^{-1}]

The choice of output weighting matrices Q_y and F_y only penalizes the biomass. The prediction and control horizons are obtained by trial and error. The convergence of the controller is influenced by the choice of transformation matrix Q_y into Q_ψ, either using equation (8) or (9). For datasets with low noise (the first scenario), the algorithm performs better if the transformation comes from the first option. The second option achieves better

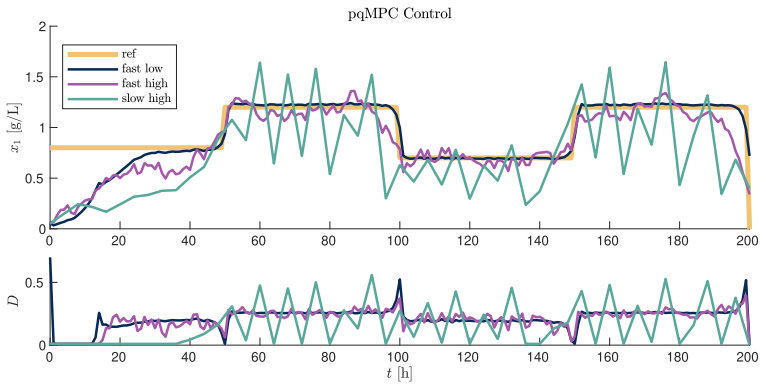

Figure 2: Linear Model Predictive Control for the three scenarios.

results for the experiment with a high noise level.

We observe a deviation from the setpoint in the scenario with a high noise level (notice the performance of the tests with high noise levels in Figure 2). This error is caused by the noise because, at every optimization step, it is necessary to know the evaluation of the function space, i.e., $\Psi(x(k))$ in (17). To obtain this value, it is required to *lift* the previous output with the set of observables, an output that has been corrupted with noise. Therefore, the next step in improving the algorithm is to develop a state observer to work in the function space and update the value of the function space at each optimization step.

CONCLUSIONS

This paper implements a complete data-driven workflow for controlling a simple bioprocess. It uses the pqEDMD algorithm to model the dynamics and a traditional MPC controller adapted to work on the pqEDMD's function space. The main methodological developments are the application of a singular value decomposition to the regression matrix to get the effective rank and improve the approximation of the ordinary least squares. This study also shows how to adapt the traditional model predictive control formulation to work with the particularities of the function space of the pqEDMD. This study shows that EDMD approximations can be interesting alternatives to neural networks, as they provide a linear system representation in a function space where linear estimators and controllers can be developed. The conditions to satisfy for a successful implementation are the convergence of the pqEDMD to a stable solution and the feasibility of the MPC solutions, which depend on the information content of the training data and the selection of tuning parameters in the pqEDMD algorithm and the MPC.

REFERENCES

1. Koopman BO. Hamiltonian Systems and Transformation in Hilbert Space. Proceedings of the National Academy of Sciences. 1931;17:315–8
2. Budišić M, Mohr R, Mezić I. Applied Koopmanisma. Chaos: An Interdisciplinary Journal of Nonlinear Science. 2012;22:047510.
3. Korda M, Mezić I. Linear Predictors for Nonlinear Dynamical Systems: Koopman Operator Meets Model Predictive Control. Automatica. 2018;93:149–60.
4. Korda M, Mezić I. On convergence of extended dynamic mode decomposition to the koopman operator. Journal of Nonlinear Science. 2018;28:687–710.
5. Williams MO, Kevrekidis IG, Rowley CW. A Data-Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition. Journal of Nonlinear Science. 2015;25:1307–46.
6. Schmid PJ. Dynamic mode decomposition of numerical and experimental data. Journal of Fluid Mechanics. 2010;656:5–28.
7. Garcia-Tenorio C, Vande Wouwer A. A matlab toolbox for extended dynamic mode decomposition based on orthogonal polynomials and p-q quasi-norm order reduction. Mathematics. 2022;10.
8. Otálora P, Guzmán J.L., Gil J.D., Berenguel M., Acién F.G., Data-driven Model Predictive Control for pH regulation in Raceway Reactors, IFAC-PapersOnLine, 2023;56:2, P. 6223-6228
9. Kouvaritakis B, Cannon M. Model predictive control: Classical, robust and stochastic. 1st ed. Springer; 2015.