Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 56-2 (2023) 10077–10082

Data-driven Model Predictive Control with Matrix Forgetting Factor

Horacio M. Calderón [*,**] **Erik Schulz** [*,**] **Thimo Oehlschlägel** [*] **Herbert Werner** [**]

[*] *IAV GmbH, Research, Gifhorn, Germany (e-mail: horacio.martinez.calderon@iav.de).*

[**] *Hamburg University of Technology, Institute of Control Systems, Hamburg, Germany*

Abstract: In this paper, a model predictive control (MPC) based on the Koopman operator framework is presented. This framework, allows to obtain a model solely from data. Moreover, it can be easily updated using a recursive least squares (RLS) approach. This aids the controller to deal with inaccuracies in the model parameters. The performance of the update mechanism can be enhanced by the inclusion of forgetting techniques. One of these is the addition of a constant forgetting factor (CFF) to the update equations. However, this technique can be very sensitive to noise and if the measured signals are not persistently exciting, numerical issues can occur. To avoid the latter, the matrix forgetting factor (MFF) presented in (Bruce et al., 2020) is used. This approach combines two forgetting techniques: variable-rate and variable-direction forgetting, which leads to a higher closed loop performance of the proposed controller.

Keywords: Data-driven, recursive least squares, Koopman operator, model predictive control.

1. INTRODUCTION

One of the most compelling model-based control schemes is model predictive control. The main reason behind this, is its ability to explicitly deal with constraints. Moreover, it is straightforward to design controllers for systems with multiple inputs and outputs (Rossiter, 2003). The MPC approach involves the solution of an optimal control problem (OCP) at each sampling instant. Whereby, the OCP relies on a model of the plant to be controlled to compute the input that achieves the goals set by the designer. The closed loop performance of an MPC depends heavily on the model, since model inaccuracies will lead to sub-optimal performance. Predictive controllers can be designed using either linear or nonlinear models. However, the computation time required for the latter is higher due to the algorithms' complexity (Gros et al., 2016). The difference in computation time has led to the development of different approaches to design fast MPC schemes for nonlinear plants (e. g. Cisneros et al. (2016), Korda and Mezić (2018), Verschueren et al. (2021)). In this paper, the predictive controller presented in (Cisneros et al., 2020) is used. This controller employs the Koopman operator framework (Budišić et al., 2012) to obtain a model of the plant through data. This is achieved by using the extended dynamic mode decomposition algorithm presented in (Williams et al., 2015). This algorithm solves a least squares problem to obtain an approximation of the Koopman operator, which in turn allows us to obtain a model. Moreover, it is possible to use RLS to update the model online (Zhang et al., 2019; Calderón et al., 2021). Furthermore, it is possible to use forgetting techniques that allow to learn changing parameters rapidly. These techniques are often used in system identification (Ljung, 1999) and adaptive control (Goodwin and Sin, 2014). One of these techniques, is the addition of a CFF. Even though its implementation is simple, it can be very sensitive to the choice of the forgetting factor. In addition, if the measured signals used for the update are not persistently exciting bursting phenomena can occur (Anderson, 1985). To circumvent this problem, many methods have been proposed, some examples are: covariance resetting, directional forgetting, bounded covariance (cf. Goel et al. (2020) and references therein). In this paper, the matrix forgetting factor approach proposed in Bruce et al. (2020) is used. This technique combines two schemes, variable-rate and variable-direction forgetting.

The data-driven controller proposed here, combines an MPC scheme based on the Koopman operator with an online update mechanism based on the MFF. As a result, the controller is able to learn changes in the model parameters fast, and if the persistence of excitation of the measured signals is lost, the closed loop performance does not deteriorate. There exist several other data-driven MPC schemes. For example, the data-enabled predictive control (Coulson et al., 2019) for linear time invariant (LTI) systems, or for nonlinear systems the data-driven predictive scheme presented in (Berberich et al., 2022). However, these assume that the measured signals are persistently exciting, which may not be the case (e. g. set point tracking).

The outline of the paper is given as follows: Section 2 gives an introduction to the Koopman operator framework. Section 3 introduces the equations that are used for the online update of the model. Section 4 describes the MPC algorithm used in this paper. Moreover, it illustrates its integration to the online update scheme. Section 5 presents

10.1016/j.ifacol.2023.10.877

the results of two simulation experiments. These test the ability of the proposed controller to deal with two issues: the loss of persistency excitation, and the noise corruption of the measured signals. Finally, Section 6 gives a conclusion and an outlook for future work.

Throughout the paper the following notation is used. Denote the transpose of a matrix A as $A^\top$, and an identity matrix of size n-by-n as I_n. A positive semi-definite matrix A is denoted as $A \succeq 0$, and a positive definite matrix B as $B \succ 0$. Denote diagonal matrix with entries $a, b, ...$ along the diagonal as $\text{diag}(a, b, ...)$. The weighted two-norm $x^\top Q x$ is denoted as $\|x\|_Q^2$. The Moore-Penrose pseudo-inverse of a matrix A is denoted as $A^\dagger$. The Frobenius norm is denoted as $\|\cdot\|_\text{F}$. The Kronecker product of the matrices A and B is denoted as $A \otimes B$. The vector representation of a matrix A is denoted as $\text{col}(A) = A_\text{v}$ (i. e. the rows of A are transposed and then stacked on a column vector A_v) The singular value decomposition of a matrix A is given by $[U, \Sigma, V] = \text{svd}(A)$ and its maximum singular value by $\sigma_{\max}(A)$.

2. PRELIMINARIES

In this section, an introduction to the Koopman operator framework is given. First, the operator is defined. Then, a description of its approximation is presented. Consider the following discrete-time nonlinear system

$$x_{k+1} = f(x_k, u_k), \tag{1}$$

where the state and the input at time instant $k \in \mathbb{Z}$ are $x_k \in \mathbb{R}^n$ and $u_k \in \mathbb{R}^m$, respectively. The function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ shifts the state forward in time. Now, define a scalar valued function of the state and the input as $\psi : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, and consider that it belongs to an infinite dimensional vector space $\mathcal{F}$. The Koopman operator acts on this function space as follows

$$[\mathcal{K}\psi](x_k, u_k) = \psi(f(x_k, u_k), h(x_k, u_k)) \; \forall \psi \in \mathcal{F},$$

where the function $h(x_k, u_k)$ shifts the input forward in time. Therefore, the operator $\mathcal{K}$ acts on $\psi(x_k, u_k)$ by moving it forward in time. In literature, the function ψ is often referred as an *observable* of the system (1). This definition of the Koopman operator was first presented in (Proctor et al., 2018), and it generalizes the definition in (Budišić et al., 2012), which does not include inputs. The definition of $h(x_k, u_k)$ depends on the type of input used, for example, if u_k is a state feedback controller then $h(x_k, u_k) = Ff(x_k, u_k)$, where F is the controller gain.

Given that $\mathcal{F}$ is a vector space, the operator $\mathcal{K}$ is linear, even though the underlying system is nonlinear. On the upside, this property makes the Koopman operator very attractive, since one can use tools from linear systems theory to analyze (Mauroy and Mezić, 2016), identify (Mauroy and Goncalves, 2019), and control (Korda and Mezić, 2018) nonlinear systems. On the downside, notice that $\mathcal{K}$ is an infinite dimensional operator. Thus, it cannot be used directly for practical purposes. Therefore, an approximation of the operator has to be made. To this end, consider a vector valued function of observables $\Psi : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n_\psi}$. The approximation of the Koopman operator $K \in \mathbb{R}^{n_\psi \times n_\psi}$ acts on $\Psi(x_k, u_k)$ by shifting it forward in time, thus we have

$$\Psi(x_{k+1}, u_{k+1}) = K\Psi(x_k, u_k) + \xi_{k+1}, \tag{2}$$

where ξ_{k+1} represents the approximation error. There are different ways to compute K. In this paper, we use the data-driven approach presented by Williams et al. (2015), which is called extended dynamic mode decomposition (EDMD). This algorithm requires two key elements for its implementation: a vector of observables Ψ and a data set. The observables can be selected in different ways, if a model of the underlying system is given, observables can be selected based on it (Calderón et al., 2022). If no model is given, general basis functions can be used (e. g. monomials, radial basis functions), or even neural networks can be used to learn the observable functions (Yeung et al., 2019). The data set consists of state and input pairs sampled along a trajectory generated by (1), i. e.

$$\mathcal{D} = \{(x_1, u_1), \dots, (x_M, u_M)\}, \tag{3}$$

where M is the number of samples. From the data set $\mathcal{D}$ the matrices

$$Y_\text{p} = [\Psi(x_1, u_1), \dots, \Psi(x_{M-1}, u_{M-1})],$$
$$Y_\text{f} = [\Psi(x_2, u_2), \dots, \Psi(x_M, u_M)]$$

can be obtained. The data matrices $Y_\text{p}, Y_\text{f} \in \mathbb{R}^{n_\psi \times M-1}$ contain the evaluations of the observable function Ψ along the data set $\mathcal{D}$. To compute K the algorithm solves the optimization problem

$$\min_K \|Y_\text{f} - KY_\text{p}\|_\text{F}^2, \tag{4}$$

the solution of this problem is given by $K^* = Y_\text{f} Y_\text{p}^\dagger$. Note that the objective of (4) is to minimize the approximation error ξ at each sampling instant of the data set $\mathcal{D}$. If the functions within Ψ form a Koopman invariant subspace and the data set is rich enough this error should be zero (cf. Brunton et al. (2016)).

3. ONLINE UPDATE WITH MATRIX FORGETTING

In this section, the recursive equations that allow the online update of the Koopman operator approximation are presented. To this end, the optimization problem (4) is reformulated, such that classical RLS methods are applicable. Then, the equations for the CFF are introduced. Finally, an introduction to the MFF is given.

Consider (2), and notice that it can be written as

$$\Psi_{k+1} = \phi_k \mathfrak{K} + \xi_{k+1}, \tag{5}$$

where the approximation of the Koopman operator is $\mathfrak{K} = \text{col}(K) \in \mathbb{R}^{n_\psi^2}$, the matrix $\phi_k = I_{n_\psi} \otimes \Psi_k^\top$ is the regressor matrix, the vector of observables at time instant k and $k+1$ are Ψ_k and Ψ_{k+1}. To have a more compact representation, the dependency of Ψ on x_k and u_k is omitted. Using (5), it is possible to write the prediction error for the entire data set (3) as

$$\begin{bmatrix} \xi_2 \\ \vdots \\ \xi_M \end{bmatrix} = \underbrace{\begin{bmatrix} \Psi_2 \\ \vdots \\ \Psi_M \end{bmatrix}}_{\mathcal{Y}_\text{f}} - \underbrace{\begin{bmatrix} I_{n_\psi} \otimes \Psi_1^\top \\ \vdots \\ I_{n_\psi} \otimes \Psi_{M-1}^\top \end{bmatrix}}_{\Phi_\text{p}} \mathfrak{K},$$

where the vector of future observables is $\mathcal{Y}_\text{f} \in \mathbb{R}^{(M-1)n_\psi}$, the regressor matrix is $\Phi_\text{p} \in \mathbb{R}^{(M-1)n_\psi \times n_\psi^2}$. Using the previous equation, the optimization problem (4) can be re-formulated as

$$\min_{\mathfrak{K}} \|\mathcal{Y}_\text{f} - \Phi_\text{p}\mathfrak{K}\|_2^2, \tag{6}$$

where the solution is $\mathfrak{K}^* = \Phi_{\mathrm{p}}^{\dagger}\mathcal{Y}_{\mathrm{f}}$. In this form, the optimization problem has the same structure as least squares problems that appear in system identification and adaptive control literature. Furthermore, one can solve this problem in a recursive manner and use forgetting techniques that aid the adaptation capabilities of the system. To see how forgetting techniques work, consider the following cost function

$$J_k(\mathfrak{K}) = \sum_{i=0}^{k} \lambda^{k-i}\|\Psi_{i+1} - \phi_i\mathfrak{K}\|^2 + \lambda^{k+1}\|\mathfrak{K} - \mathfrak{K}_0\|_Q^2 \quad (7)$$

where the $\lambda \in (0,1]$ is the forgetting factor, $\mathfrak{K}_0 \in \mathbb{R}^{n_\psi^2}$ is the initial value of the approximation of the Koopman operator, and $Q \succ 0$ weights the difference between the initial and current approximation. Notice that by removing the cost on the initial guess for the parameters and setting $\lambda = 1$ the cost function in (6) is recovered. If the value of λ is less than one, the cost function puts more weight on recent values. Therefore, improving the ability to learn parameters that change in time. Hence, the name forgetting factor, since it allows to forget the older parameters. Consider $k \geq 0$, and define $P_0 = Q^{-1}$, the recursive equations that minimize (7) are

$$\nu_k = P_k\phi_k^\top(\lambda I_{n_\psi} + \phi_k P_k \phi_k^\top)^{-1}, \quad (8a)$$
$$\mathfrak{K}_{k+1} = \mathfrak{K}_k + \nu_k(\Psi_{k+1} - \phi_k\mathfrak{K}_k), \quad (8b)$$
$$P_{k+1} = \frac{1}{\lambda}(I_{n_\psi^2} - \nu_k\phi_k)P_k, \quad (8c)$$

where $\nu_k \in \mathbb{R}^{n_\psi^2 \times n_\psi}$ is a correction matrix that weights the current prediction error. The matrix $P_k \in \mathbb{R}^{n_\psi^2 \times n_\psi^2}$ is symmetric and positive definite. It is known as the covariance matrix, this comes from the interpretation of RLS as a one step optimal predictor (cf. Goel et al. (2020)). Having big entries in P_k means that the confidence in the initial parameters' guess is low, while having small entries reflects a high confidence. Even though, equations (8) are valid to approximate the Koopman operator, the resulting approximation $\mathfrak{K}$ is a vector. Therefore, reshaping it as a matrix is needed. To avoid this operation, the equations that implement the constant forgetting factor in matrix form are used. These are

$$\gamma_k = \frac{1}{\Psi_k^\top\Gamma_k\Psi_k + \lambda}\Psi_k^\top\Gamma_k, \quad (9a)$$
$$K_{k+1} = K_k + (\Psi_{k+1} - K_k\Psi_k)\gamma_k, \quad (9b)$$
$$\Gamma_{k+1} = \Gamma_k(I_{n_\psi} - \Psi_k\gamma_k)\frac{1}{\lambda}, \quad (9c)$$

where $\gamma_k \in \mathbb{R}^{1\times n_\psi}$ is a correction vector, the approximation of the Koopman operator at time instant k is $K_k \in \mathbb{R}^{n_\psi \times n_\psi}$. The matrix $\Gamma_k \in \mathbb{R}^{n_\psi \times n_\psi}$ is the covariance matrix. The derivation of these equations is presented in (Calderón et al., 2021). The effectiveness of equations (8) and (9) depends on the persistence of excitation of the measured signals. If $\lambda < 1$ and persistency condition is not met, the singular values of the covariance matrix will diverge, hence, leading to numerical issues (cf. Goel et al. (2020)). In control applications, this condition is not always met because often the task is to track a setpoint. Such task, leads to constant values of the inputs and states. Thus, loosing the persistence of excitation. To deal with this issue, the matrix forgetting factor (MFF) presented in (Bruce et al., 2020) is used. Specifically, we use the special case of MFF were variable-rate and variable-direction forgetting are achieved. Moreover, the equations that give the update are in matrix form instead of a vector of parameters. For all $k \geq 0$, let $\beta_k \in (0,\infty)$, the singular value decomposition of $\Gamma_k = U_k\Sigma_k U_k^\top$, define

$$\zeta_k = \Psi_k^\top U_k.$$

Now, for a small threshold value $\epsilon > 0$, and $i \in \{1,\dots,n_\psi\}$ let

$$\Lambda_k(i,i) = \begin{cases} \sqrt{\beta_k} & \text{if } \|\zeta_{k,i}\| > \epsilon, \\ 1 & \text{otherwise,} \end{cases}$$

where $\zeta_{k,i}$ is the i-th element of ζ_k. Then, the update equations are

$$B_k = U_k\Lambda_k U_k^\top \quad (10a)$$
$$\tilde{\Gamma}_k = B_k\Gamma_k B_k^\top \quad (10b)$$
$$\gamma_k = \frac{1}{\Psi_k^\top\tilde{\Gamma}_k\Psi_k + 1}\Psi_k^\top\tilde{\Gamma}_k, \quad (10c)$$
$$K_{k+1} = K_k + (\Psi_{k+1} - K_k\Psi_k)\gamma_k, \quad (10d)$$
$$\Gamma_{k+1} = \tilde{\Gamma}_k(I_{n_\psi} - \Psi_k\gamma_k). \quad (10e)$$

Finally, β_k is computed as

$$\beta_k = \begin{cases} 1 + \eta\,\mathrm{sat}_\delta(E_W) & \text{if } E_W > \alpha, \\ 1 & \text{if } E_W \leq \alpha, \end{cases}$$

where $\alpha > 0$, $\eta > 0$, $\delta > 0$ are tuning knobs, the function sat_δ is the saturation function with saturation level δ and

$$E_W = \sqrt{\frac{1}{W}\sum_{i=k-W}^{k}\|\Psi_{i+1} - K_k\Psi_i\|^2},$$

where W is the number of samples used to compute E_W. Note that the MFF has five tuning knobs. These can be divided into two categories: variable-rate forgetting, and variable-direction forgetting. The variable-rate forgetting has the tuning knobs W, α, η, and δ. The parameter W serves as a window size to compute the average approximation error E_W. The parameter α is a threshold that decides when to change the forgetting factor. Finally, η is a scaling and δ a saturation limit used to compute the forgetting factor. Note that the forgetting factor is $\lambda = 1/\beta_k$ (cf. Bruce et al. (2020)). The variable-direction forgetting only has the parameter ϵ, and it determines if enough information content has been received in the i-th information direction. To recognize this, consider each column of U_k as an information direction. Thus, ζ_k is the projection of the measurement Ψ_k onto the information directions. Therefore, the information content along i-th information direction is given by the norm of the i-th element of ζ_k. This allows the algorithm to recognize in which direction a forgetting factor less than one is allowed, since otherwise it would lead to the growth of the singular value corresponding to that direction. The parameter ϵ should be selected to be larger than the noise-to-signal ratio or larger than the machine zero in a noise free simulation.

4. CONTROL DESIGN

In this section, an approach to design controllers using the Koopman operator is presented. We rely on the predictive approach proposed in (Cisneros et al., 2020). This choice comes from the amount of processing that is needed to update the model online, which for other approaches is higher (cf. Calderón et al. (2021)).

To start, it is necessary to obtain a model from the approximation of the Koopman operator K. Consider an observable function of the form $\Psi_k^\top = [x_k^\top, \psi_{n_h}^\top(x_k, u_k), u_k^\top]$ where $\psi_{n_h} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n_h}$ is a vector valued function containing nonlinear functions of the states and inputs. The evolution of the state can be written as

$$x_{k+1} \approx \hat{K}\Psi_k,$$

where $\hat{K} = [I_n, 0, 0]K$. Using the velocity-based linearization it is possible to obtain an approximation of the state increment as

$$\Delta x_{k+1} \approx \hat{K}\frac{\partial \Psi_k}{\partial x_k}\bigg|_{x_k,u_k} \Delta x_k + \hat{K}\frac{\partial \Psi_k}{\partial u_k}\bigg|_{x_k,u_k} \Delta u_k,$$

with state and input increments Δx_k and Δu_k. From this approximation it is possible to obtain the quasi-linear parameter-varying (LPV) model

$$\begin{bmatrix} x_{k+1} \\ \Delta x_{k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} I_n & \hat{K}\dfrac{\partial \Psi_k}{\partial x_k} \\ 0 & \hat{K}\dfrac{\partial \Psi_k}{\partial x_k} \end{bmatrix}}_{A(\rho_k)} \begin{bmatrix} x_k \\ \Delta x_k \end{bmatrix} + \underbrace{\begin{bmatrix} \hat{K}\dfrac{\partial \Psi_k}{\partial u_k} \\ \hat{K}\dfrac{\partial \Psi_k}{\partial u_k} \end{bmatrix}}_{B(\rho_k)} \Delta u_k,$$

where $\rho_k = H[x_k^\top, u_k^\top]^\top$, and H is a matrix that selects the appropriate states and inputs that schedule the model. Defining the state as $z_k^\top = [x_k^\top, \Delta x_k^\top]$, the model can be written in the compact form

$$z_{k+1} = A(\rho_k)z_k + B(\rho_k)\Delta u_k. \tag{11}$$

Model predictive control strategies solve at each sampling instant an OCP. The OCP consists of three main components: the cost, the model, and the constraints. The cost provides the designer with the tuning knobs to shape the closed loop performance of the system. The model represents the plant to be controlled. The model plays a major role in MPC strategies, since the higher the accuracy the better the predictions. Having good predictions aids the controller to make better decisions, thus, leading to a better performance. The constraints allow the designer to add limitations to what the controller is allowed to do. For example, it is possible to add limits on the inputs, which are ubiquitous in the control of real systems.

The proposed control approach solves the following OCP

$$\min_{\Delta U_k} \ \Phi(e_{k+N}) + \sum_{i=0}^{N-1} \|e_{k+i}\|_T^2 + \|\Delta u_{k+i}\|_R^2 \tag{12a}$$

subject to

$$z_{k+j+1} = A(\rho_{k+j})z_{k+j} + B(\rho_{k+j})\Delta u_{k+j}, \tag{12b}$$

$$u_{k+j} = u_{k-1} + \sum_{i=0}^{j}(\Delta u_{k+i}) \in \mathcal{U}, \tag{12c}$$

$$z_{k+j} \in \mathcal{Z} \text{ for } j \in \{0, \ \ldots, \ N\},$$

where the vector $\Delta U_k^\top = [\Delta u_k^\top, \ldots, \Delta u_{k+N-1}^\top]$ represents the decision variables, the prediction horizon is N, the tracking error is $e_k = r_k - z_k$, the matrices $T \succeq 0$ and $R \succ 0$ are tuning knobs weighting the tracking error and control effort. The terminal weight is $\Phi(e_{k+N}) = \|e_{k+N}\|_T^2$. The sets $\mathcal{U}$ and $\mathcal{Z}$ represent input and state constraint sets. Note that (12) is non-convex because the model constraint is nonlinear. However, if we fix the trajectory of scheduling parameter $\mathcal{P}_k^\top = [\rho_k^\top, \ldots, \rho_{k+N-1}^\top]$ and consider only linear constraints, then the problem is convex (Cisneros et al., 2016). In fact, the optimization problem is a quadratic optimization problem (QP), which can be solved efficiently. Note that at each sampling instant k the OCP (12) has to be reformulated into a QP, which allows us to update model (12b) using the CFF equations (9) or the MFF equations (10). A pseudo algorithm of the predictive control strategy with online update is given in Algorithm 1.

Algorithm 1. Koopman quasi-LPV MPC with online update.

```
// Initialize controller parameters
T, R, N, Z, U.
// Initialize update parameters
update-flag, K_k = K_0, Γ_k = (Y_p Y_p^⊤)^{-1}, λ, W, α, δ, η, ϵ
for k = 0 to T_sim do
    Read x_k, x_{k-1} and u_{k-1}
    if update-flag is CFF then
        K_k ← Update K using (9)
    else
        if W samples of x and u were collected then
            // Variable-rate forgetting
            Compute E_W and β_k
            // Variable-direction forgetting
            Compute [U_k, Σ_k, V_k] = svd(Γ_k)
            Compute ζ_k and Λ_k
            K_k ← Update K using (10)
        end
    end
    Using K_k formulate the OCP (12) as a QP.
    u_k ← Solve the QP.
    Apply u_k to the system.
end
```

5. NUMERICAL EXAMPLE

In this section, simulation experiments that showcase the advantages and disadvantages of a Koopman operator-based quasi-LPV model predictive control (KqLMPC) controller using the CFF and the MFF are presented. The controllers are referred to as KqLMPC-CFF and KqLMPC-MFF to distinguish which forgetting technique they use. The simulation experiment consists in the control of a nonlinear system with a sudden change in its parameters. The control task is to track a series of step changes. Case scenarios with and without measurement noise are presented. The case without noise is used to show the effect of the persistence of excitation of the measurement signals. The case with noise is used to evaluate the sensitivity of the controllers to noise.

The plant to be controlled is a Van der Pol oscillator with a time varying parameter. The dynamics are given by differential equation

$$\ddot{x} + \mu(t)(x^2 - 1)\dot{x} + x = u, \tag{13}$$

where x is the position and u is the input. The parameter μ is a piecewise function given by

$$\mu(t) = \begin{cases} 1 & \text{if } t \leq 10, \\ 3 & \text{if } t > 10. \end{cases}$$

To perform the experiments an initial approximation of Koooopman operator is computed using EDMD. To this end, an open loop simulation experiment of the Van der Pol oscillator starting with initial conditions $\dot{x}_0 = x_0 = 5$ is

carried out. The experiment only considers the dynamics when the value of the parameter $\mu = 1$. The system is excited using band limited white noise. It is assumed that the position and velocity are measured with a sampling rate of $T_s = 10$ms. The observable function is defined as

$$\Psi_k^\top = \left[x_k,\ v_k,\ \psi_{n_\psi}^\top(x_k, v_k),\ u_k\right],$$

where x_k is the position, v_k is the velocity, and u_k is the input. The function $\psi_{n_\psi}(x_k, v_k)$ is selected as unique monomial combinations of the position and the velocity up to degree three, this yields $n_\psi = 10$. This selection provides good trade-off between prediction accuracy and the number of observables. The controllers are tuned to reach the reference in a short period of time with little overshoot. For the case scenario without noise, the tuning parameters are $N = 40$, $T = \text{diag}(750, 40, 0, 0)$, $R = 0.75$, for the case with noise the parameters are $N = 40$, $T = \text{diag}(300, 30, 0, 0)$, $R = 50$. This difference allows the controllers to be less aggressive, and thus deal with the noise better. Note that in both controllers the tuning of the MPC algorithms is the same. Thus, the difference in performance comes from the forgetting technique used.

The tuning knobs for the forgetting techniques for the case scenario without noise are: for the KqLMPC-CFF $\lambda = 0.978$, and for the KqLMPC-MFF $W = 5$, $\alpha = 0.015$, $\eta = 10$, $\delta = 1000$, $\epsilon = 10^{-14}$. The results of the case scenario without noise are depicted in Figure 1. The performance of the controllers is almost identical before the parameter change. After the parameter change the difference of the controllers becomes evident. Notice that the response of the KqLMPC-CFF has over- and undershoots, while for the KqLMPC-MFF the response remains the same. This difference comes from the update scheme that the controllers use. On the one hand, the KqLMPC-CFF has constant forgetting factor and it is less than one, this may cause trouble if the measured signals are not persistently exciting (cf. Goel et al. (2020)), which is the case in this simulation experiment. On the other hand, the KqLMPC-MFF detects when a value of $\lambda < 1$ is needed and it is changed accordingly. Furthermore, if the signal is not persistently exciting the MFF changes Λ_k such that the singular values of the covariance matrix Γ_k do not diverge. In Figure 3, the maximum singular values corresponding to this simulation experiment are presented. Notice that for the periods of time when the measured signal is constant, the singular values of the covariance matrix increase for the KqLMPC-CFF, while for the KqLMPC-MFF the singular values do not increase.

The tuning knobs for the forgetting schemes for the case scenario with noise are: for the KqLMPC-CFF $\lambda = 0.999$, and for the KqLMPC-MFF $W = 5$, $\alpha = 0.065$, $\eta = 1$, $\delta = 10$, $\epsilon = 10^{-4}$. The results of the case scenario with noise are depicted in Figure 2. Here, the measured position and velocity are corrupted by zero-mean Gaussian white noise with standard deviation of 0.01. Again, the performance of the controllers before the parameter change is almost identical. However, after the change the performance of the KqLMPC-CFF degrades slightly, there is an overshoot, and when the reference changes from one to zero the response is slightly slower than the KqLMPC-MFF. Notice that the forgetting factor of the KqLMPC-MFF changes

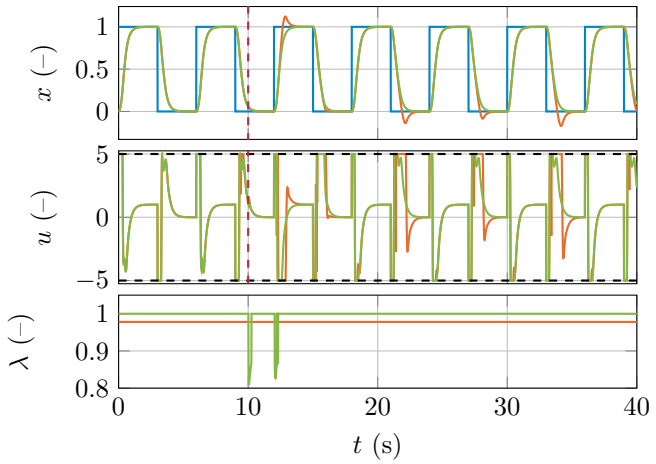

Figure 1. Case scenario without noise. Reference (—), KqLMPC-CFF (—), KqLMPC-MFF (—) parameter change (- - -), input constraints (- - -).

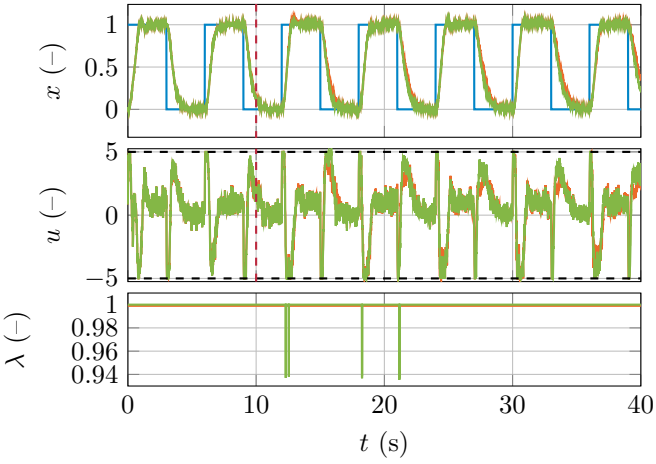

Figure 2. Case scenario with noise. Reference (—), KqLMPC-CFF (—), KqLMPC-MFF (—) parameter change (- - -), input constraints (- - -).

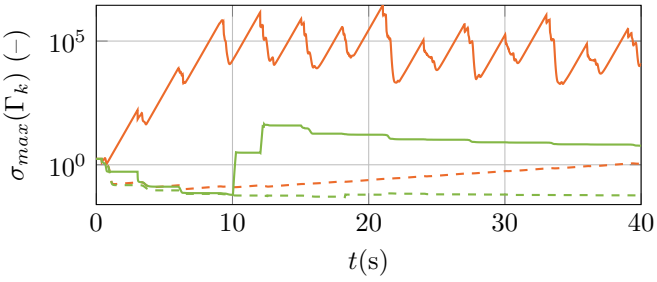

Figure 3. Maximum singular value of the matrix Γ_k. KqLMPC-CFF with noise (- - -) without noise (—), KqLMPC-MFF with noise (- - -), without noise (—).

after the parameter change, and its value is bigger than the case without noise. This, comes from the difference in tuning, which is needed to deal with the noise. The maximum singular values of the covariance matrices for this case scenario are depicted in Figure 3. In this case, the values of the covariance stay way below since the added measurement noise provides a persistently exciting signal.

Table 1. Algorithms' execution time (in ms).

	ET		UT		MCT		OCPST	
	Mean	Max.	Mean	Max.	Mean	Max.	Mean	Max.
CFF	1.078	7.663	0.011	0.185	0.724	2.984	0.139	6.460
MFF	1.084	8.267	0.062	0.596	0.704	2.653	0.119	6.250

An important aspect of an MPC control strategy is its execution time (ET), since it determines if it is implementable on embedded hardware or not. For this reason, the time that takes to execute the controllers was measured for the case where no noise was added. The execution time is divided in three components: the update time (UT) which is the time used to update K, the matrix construction time (MCT) which is the time needed to construct the matrices used for the QP formulation of the OCP, and the OCP solution time (OCPST) which is the time that takes for the solver to solve the QP. Table 1 presents these values. Notice that the only component that presents a big difference is the update time. This comes from the forgetting technique that is used. The MFF has a more complex update mechanism, since it involves the singular value decomposition of the covariance matrix Γ_k. It takes on average almost six times more time to update K using the MFF than using the CFF.

6. CONCLUSION & OUTLOOK

In this paper, a Koopman operator-based MPC strategy that incorporates the matrix forgetting factor as way to update its model was presented. This technique was compared to the constant forgetting factor. The comparison was carried out in two case scenarios: one where there was a lack of persistency of excitation of the measurement signals, and one where noise was added. As a result, two advantages of the proposed scheme were recognized. First, the lack of persistency of excitation did not affect the performance of the controller. Second, the controller presented better performance when noise is present because it could adapt faster to the parameter change. However, these advantages come at the price of computational complexity, thus, the proposed scheme required more time for its execution. As an outlook, conditions for stability and performance guarantees have to be studied.

REFERENCES

Anderson, B.D. (1985). Adaptive systems, lack of persistency of excitation and bursting phenomena. *Automatica*, 21(3), 247–258.

Berberich, J., Kohler, J., Muller, M.A., and Allgower, F. (2022). Linear tracking MPC for nonlinear systems—part II: The data-driven case. *IEEE Transactions on Automatic Control*, 67(9), 4406–4421.

Bruce, A.L., Goel, A., and Bernstein, D.S. (2020). Recursive least squares with matrix forgetting. In *2020 American Control Conference (ACC)*. IEEE.

Brunton, S.L., Brunton, B.W., Proctor, J.L., and Kutz, J.N. (2016). Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PLOS ONE*, 11(2), e0150171.

Budišić, M., Mohr, R., and Mezić, I. (2012). Applied koopmanism. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(4), 047510.

Calderón, H.M., Schulz, E., Oehlschlägel, T., and Werner, H. (2021). Koopman operator-based model predictive control with recursive online update. In *2021 European Control Conference (ECC)*, 1543–1549. IEEE.

Calderón, H.M., Hammoud, I., Oehlschlagel, T., Werner, H., and Kennel, R. (2022). Data-driven model predictive current control for synchronous machines: a koopman operator approach. In *2022 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM)*. IEEE.

Cisneros, P.S.G., Datar, A., Göttsch, P., and Werner, H. (2020). Data-driven quasi-lpv model predictive control using koopman operator techniques. In *21st IFAC World Congress*.

Cisneros, P.S.G., Voss, S., and Werner, H. (2016). Efficient nonlinear model predictive control via quasi-LPV representation. In *IEEE 55th Conference on Decision and Control*.

Coulson, J., Lygeros, J., and Dorfler, F. (2019). Data-enabled predictive control: In the shallows of the DeePC. In *2019 18th European Control Conference*. IEEE.

Goel, A., Bruce, A.L., and Bernstein, D.S. (2020). Recursive least squares with variable-direction forgetting: Compensating for the loss of persistency [lecture notes]. *IEEE Control Systems*, 40(4), 80–102.

Goodwin, G.C. and Sin, K.S. (2014). *Adaptive filtering prediction and control*. Courier Corporation.

Gros, S., Zanon, M., Quirynen, R., Bemporad, A., and Diehl, M. (2016). From linear to nonlinear MPC: bridging the gap via the real-time iteration. *International Journal of Control*, 93(1), 62–80.

Korda, M. and Mezić, I. (2018). Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93, 149–160.

Ljung, L. (1999). *System identification: Theory for the user*. Prentice-Hall, Inc.

Mauroy, A. and Mezić, I. (2016). Global stability analysis using the eigenfunctions of the koopman operator. *IEEE Transactions on Automatic Control*, 61(11), 3356–3369.

Mauroy, A. and Goncalves, J. (2019). Koopman-based lifting techniques for nonlinear systems identification.

Proctor, J.L., Brunton, S.L., and Kutz, J.N. (2018). Generalizing koopman theory to allow for inputs and control. *SIAM Journal on Applied Dynamical Systems*.

Rossiter, J.A. (2003). *Model-based predictive control: a practical approach*. CRC press.

Verschueren, R., Frison, G., Kouzoupis, D., Frey, J., van Duijkeren, N., Zanelli, A., Novoselnik, B., Albin, T., Quirynen, R., and Diehl, M. (2021). acados – a modular open-source framework for fast embedded optimal control. *Mathematical Programming Computation*.

Williams, M.O., Kevrekidis, I.G., and Rowley, C.W. (2015). A data–driven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6), 1307–1346.

Yeung, E., Kundu, S., and Hodas, N. (2019). Learning deep neural network representations for koopman operators of nonlinear dynamical systems. In *2019 American Control Conference (ACC)*. IEEE.

Zhang, H., Rowley, C.W., Deem, E.A., and Cattafesta, L.N. (2019). Online dynamic mode decomposition for time-varying systems. *SIAM Journal on Applied Dynamical Systems*, 18(3), 1586–1609.