2022 American Control Conference (ACC)
Atlanta, USA, June 8-10, 2022

Physics-Guided Neural Networks for Feedforward Control: An Orthogonal Projection-Based Approach

Johan Kon[1], Dennis Bruijnen[2], Jeroen van de Wijdeven[3], Marcel Heertjes[1,3], and Tom Oomen[1,4]

Abstract— **Unknown nonlinear dynamics can limit the performance of model-based feedforward control. The aim of this paper is to develop a feedforward control framework for systems with unknown, typically nonlinear, dynamics. To address the unknown dynamics, a physics-based feedforward model is complemented by a neural network. The neural network output in the subspace of the model is penalized through orthogonal projection. This results in uniquely identifiable model coefficients, enabling increased performance and similar task flexibility with respect to the model-based controller. The feedforward framework is validated on a representative system with performance limiting nonlinear friction characteristics.**

I. Introduction

Feedforward control is a method to significantly improve the performance of dynamic systems [1]–[4]. In feedforward control, both high performance and task flexibility are desired, i.e., a small tracking error for a variety of references. To realize task flexibility, the feedforward signal is parametrized as the output of a filter driven by a reference [5]. It is often desired that this filter is interpretable, i.e., that it has physically meaningful coefficients. Perfect feedforward is guaranteed if the feedforward filter is an accurate description of the inverse dynamics of the system.

Classically, the feedforward controller is parametrized based on physical insights. For example, the inverse dynamics can be explicitly parametrized as a polynomial or rational transfer function [6], for which the coefficients can be tuned by hand or learned through data-based methods, e.g., iterative learning control [7]. Alternatively, the original system can be identified using system identification tools [8] and inverted accordingly [9]–[11]. However, this parametrization of the feedforward controller based on physical insights limits the achievable performance in the presence of unknown dynamics [7], and may result in parameter bias.

To overcome the downsides of the physics-based parametrization in the context of unknown, typically nonlinear dynamics, neural networks have been used in feedforward control as a more flexible parametrization to enhance performance [12]–[14]. Neural networks can provide a rich parametrization of the feedforward controller due to their universal approximator characteristics [15], and enable compensation of unknown (nonlinear) dynamics accordingly. However, the dynamics captured in neural networks often lack physical interpretation [16] and neural networks are known to extrapolate poorly [17].

Recently, physically meaningful parametrizations in the form of models have been combined with neural networks to combine physical insights and increased performance by learning unmodelled effects [18], [19]. This is achieved, i.a., through using physics-based model outputs as additional inputs to the neural network. For the specific choice of neural networks with residual connections [21], this corresponds to a parallel combination of a model and neural network. These parallel physics-guided neural networks (PGNN) have first been used for feedforward control in [20], in which model-based basis functions are used as features to control linear motors. While these PGNNs perform better, train faster, and show superior generalization properties when compared to their black-box counterpart, they do not distinguish well between the contributions of the model and the neural network. As a result, the interpretability and generalization are lacking compared to a fully physics-based approach.

Although some recent works appeared on using neural networks in feedforward control, there does not yet exist a framework in which the contribution of the physical model, capturing the known dynamics, and the neural network, capturing the unknown dynamics, are explicitly separated. This paper aims to develop a feedforward control framework that combines a physics-based model with a neural network in such a way that the model remains interpretable with similar task flexibility (extrapolation) and performance as a purely model-based parametrization, while the neural network increases the performance by capturing the unknown dynamics inside the training regime. The main contribution is an orthogonal projection-based cost function to ensure that the neural network compensates only the unknown dynamics.

This contribution consists of the following cornerstones. First, the physics-guided parametrization is introduced in Section II. Second, in Section III, it is shown that this parametrization combined with the least-squares cost function results in an uninterpretable model. Third, the orthogonal projection-based cost function is introduced in Section IV. Last, in Section V, the framework is exemplified on a simulated system with unknown nonlinear friction characteristics.

Notation and Definitions: All systems are discrete time, single-input single-output (SISO) with sample rate f_s. $\mathbb{Z}_{>0}$, $\mathbb{R}_{\geq 0}$ represent the set of positive integers and non-negative real numbers respectively. im A is the column space of A.

II. Problem Formulation

In this section, feedforward control for dynamic systems is introduced. Then, a physics-guided parametrization based

This work is supported by Topconsortia voor Kennis en Innovatie (TKI), and ASML and Philips Engineering Solutions. [1]: Control Systems Technology Group, Departement of Mechanical Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands, e-mail: j.j.kon@tue.nl. [2]: Philips Engineering Solutions, High Tech Campus 34, 5656 AE Eindhoven, The Netherlands. [3]: ASML, De Run 6501, 5504 DR Veldhoven, The Netherlands. [4]: Delft University of Technology, P.O. Box 5, 2600 AA Delft, The Netherlands.

978-1-6654-5197-0/$31.00 ©2022 AACC

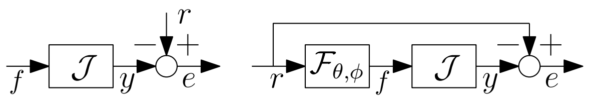


Fig. 1: Feedforward setup with input f, dynamic system $\mathcal{J}$, reference r, and error e (left). The input f is parametrized as the output of a reference dependent filter $\mathcal{F}_{\theta,\phi}$ (right).

on models and neural networks is introduced for feedforward control. Lastly, the learning problem is formulated.

A. Dynamic Processes and Datasets of Optimal Inputs

Consider the feedforward setup in Fig. 1 (left). The goal of feedforward control is to ensure that the output $y(k) \in \mathbb{R}$ of a discrete-time (DT) dynamic process $\mathcal{J}$ equals the desired output $r(k) \in \mathbb{R}$ such that the error $e(k) \in \mathbb{R}$, given by

$$e(k) = r(k) - y(k) = r(k) - \mathcal{J}(f(k)), \tag{1}$$

is zero $\forall k \in \mathbb{Z}_{>0}$, in which $f(k) \in \mathbb{R}$ is the input to the process, and $k \in \mathbb{Z}_{>0}$ the time index. The dynamic process $\mathcal{J}$ can represent either a closed-loop or open-loop system, and is defined as follows.

Definition 1 *The nonlinear dynamic system $\mathcal{J}$ with input $f(k) \in \mathbb{R}$ satisfies the ordinary difference equation (ODE)*

$$a^T \tilde{y}(k) + g_y\left(\tilde{y}(k)\right) = f(k), \tag{2}$$

with $\tilde{y}(k) = \left[y(k),\ y^{(1)}(k),\ \ldots,\ y^{(m)}(k)\right]^T \in \mathbb{R}^{m+1}$ the vector of the output and its $m \geq 0$ discrete-time derivatives

$$y^{(n+1)}(k) = \delta^{n+1} y(k) = f_s \left(y^{(n)}(k) - y^{(n)}(k-1)\right), \tag{3}$$

in which f_s is the sampling frequency, and $a = [a_0,\ a_1,\ \ldots,\ a_m]^T \in \mathbb{R}^{m+1}$. $g_y : \mathbb{R} \to \mathbb{R}$ is an unknown static globally Lipschitz function.

Remark 2 *Note that* (2) *is linear in input f, and no derivatives of f are present. Consequently, the input f corresponding to a sufficiently smooth desired output r can be found by evaluating* (2) *for $y = r$.*

Remark 3 *All results in this paper hold for systems that are linear in parameters (LIP) a, with unknown dynamics g_y. Without loss of generality, the linear model class is considered here as a specific case.*

For dynamic process $\mathcal{J}$, a dataset $\mathcal{D} = \{r_j, \hat{f}_j\}_{j=1}^{N_\mathcal{D}}$ of $N_\mathcal{D}$ references $r_j(k)$ with finite length N_j and corresponding optimal inputs $\hat{f}_j(k)$ is available such that

$$r_j(k) = \mathcal{J}(\hat{f}_j(k))\ \forall k \in [1, N_j]. \tag{4}$$

Note that this implicitly assumes that an input exists that leads to r_j. This dataset of optimal inputs is used to train a feedforward filter for $\mathcal{J}$ that generates the feedforward $f(k)$ based on reference $r(k)$, i.e., to identify an inverse of $\mathcal{J}$.

B. Physics-Guided Feedforward Parametrization

To obtain perfect performance, i.e., $e(k) = 0\ \forall k \in \mathbb{Z}_{>0}$, for a variety of references r, the input f is parametrized as the output of a reference-dependent feedforward filter $\mathcal{F}_{\theta,\phi}$ that encapsulates the system class (2), see Fig. 1. $\mathcal{F}_{\theta,\phi}$ consists of the parallel combination of a physics-based model $\mathcal{M}_\theta$ and a function approximator $\mathcal{C}_\phi$, i.e.,

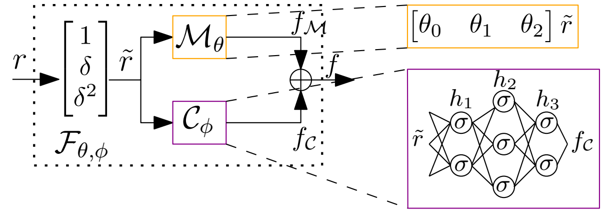


Fig. 2: Feedforward filter $\mathcal{F}_{\theta,\phi}$ with model $\mathcal{M}_\theta$ and neural network $\mathcal{C}_\phi$, here with $N_\theta = 3$ and $L = 3$ hidden layers.

$$\mathcal{F}_{\theta,\phi} : r(k) \to f(k),\ \ f(k) = \mathcal{M}_\theta(r(k)) + \mathcal{C}_\phi(r(k)). \tag{5}$$

Definition 4 (Model class) *The physics-based feedforward model $\mathcal{M}_\theta$ is parametrized as the weighted sum of the reference and its derivatives to encapsulate the known linear dynamics of* (2) *according to*

$$\mathcal{M}_\theta : r(k) \to f_\mathcal{M}(k),\ \ f_\mathcal{M}(k) = \theta^T \tilde{r}(k), \tag{6}$$

with parameter vector $\theta = [\theta_0,\ \theta_1,\ \ldots\ \theta_{N_\theta - 1}]^T \in \mathbb{R}^{N_\theta}$ and $\tilde{r}(k) = \left[r(k) \quad r^{(1)}(k) \quad \ldots \quad r^{(N_\theta - 1)}(k)\right]^T \in \mathbb{R}^{N_\theta}$.

Definition 5 (Approximator class) *The approximator $\mathcal{C}_\phi$ is parametrized as a feedforward neural network (FNN) with parameters ϕ, i.e.,*

$$\begin{aligned} &\mathcal{C}_\phi : r(k) \to f_\mathcal{C}(k), \\ &f_\mathcal{C}(k) = W_L \sigma(W_{L-1} \cdots \sigma(W_0 \tilde{r}(k) + b_0) + b_{L-1}), \end{aligned} \tag{7}$$

in which $\sigma(\cdot)$ is an element-wise activation function, such as a sigmoid, hyperbolic tangent, or rectified linear unit (ReLU). The full parameter set of the approximator with L hidden layers is given by $\phi = \{W_i, b_i\}_{i=0}^{L-1} \cup \{W_L\}$, where W_i, b_i are real, appropriately sized weights and biases.

The model explicitly incorporates the prior knowledge on the structure of $\mathcal{J}$, whereas the neural network allows for learning the unmodeled dynamics g_y. For this model and approximator, the feedforward filter is given by (see Fig. 2)

$$f(k) = \theta^T \tilde{r}(k) + \mathcal{C}_\phi(r(k)). \tag{8}$$

Remark 6 *The parametrization of $\mathcal{F}_{\theta,\phi}$ as a parallel linear and nonlinear transformation on $\tilde{r}(k)$ can be interpreted as a single residual layer. This residual layer learns deviations from a linear transformation $\mathcal{M}_\theta$, which is easier than learning an approximate linear mapping over the complete input space [21].*

Given (2) with $m = N_\theta - 1$, $\theta = a$ and using $\mathcal{C}_\phi$ to approximate g_y, the feedforward filter $\mathcal{F}_{\theta,\phi}$ is indeed able to model the inverse of $\mathcal{J}$. Thus, it has the potential to generate high performance feedforward with high task flexibility, given correct estimates of θ, ϕ.

C. Problem Formulation

The aim of this paper is to learn the parameters θ, ϕ of $\mathcal{F}_{\theta,\phi}$ in (8), based on the dataset $\mathcal{D}$, such that $\theta = a$ and $g_y = g_\phi$, resulting in both an interpretable model as well as $e(k) = 0\ \forall k \in \mathbb{Z}_{>0}$. This includes

1) illustrating that a standard least squares criterion to fit $\mathcal{F}_{\theta,\phi}$ on $\mathcal{D}$ results in model coefficients θ that

cannot be uniquely determined, resulting in the lack of interpretability and poor generalization of the model,

2) regularizing the output space of the approximator to recover interpretable model coefficients θ, and
3) illustrating the proposed approach on a simulated system with Stribeck-like friction characteristics.

III. Non-Uniqueness and Implications

Consider the least-squares criterion $J_{LS} \in \mathbb{R}_{\geq 0}$ given by

$$J_{LS} = \sum_{j=1}^{N_{\mathcal{D}}} \sum_{k=1}^{N_j} \left(\hat{f}_j(k) - \mathcal{M}_\theta(r_j(k)) - \mathcal{C}_\phi(r_j(k)) \right)^2. \quad (9)$$

This criterion is often employed in literature for regression [13], [19], [20]. In this section, it is shown that combining J_{LS} with the parallel parametrization $\mathcal{F}_{\theta,\phi}$ in (8) results in an optimum $\theta^*, \phi^* = \arg\min_{\theta,\phi} J_{LS}$ that is not unique, i.e., the model coefficients θ^* are non-unique. Consequently, the model is uninterpretable. Lastly, it is illustrated that non-unique model coefficients θ^* prevent generalization.

A. Case 1: Overparametrization of the Feedforward Filter

One source of non-uniqueness of θ^* is overparametrization of the parallel filter $\mathcal{F}_{\theta,\phi}$, such that different coefficients θ, ϕ result in the same input-output (IO) behaviour, as formalized in the following definition.

Definition 7 *The parametrization $\mathcal{F}_{\theta,\phi}$ is identifiable if for two parameter tuples (θ_1, ϕ_1), (θ_2, ϕ_2) it holds that [8]*

$$\mathcal{F}_{\theta_1,\phi_1} = \mathcal{F}_{\theta_2,\phi_2} \Rightarrow (\theta_1, \phi_1) = (\theta_2, \phi_2), \quad (10)$$

in which the filter equality is defined as

$$\mathcal{F}_{\theta_1,\phi_1} = \mathcal{F}_{\theta_2,\phi_2} \Leftrightarrow \mathcal{F}_{\theta_1,\phi_1}(r) = \mathcal{F}_{\theta_2,\phi_2}(r)\ \forall r. \quad (11)$$

For the naive choice of linear activation functions, $\mathcal{C}_\phi$ reduces to an affine mapping, such that $\mathcal{F}_{\theta,\phi}$ is not identifiable, as formalized by the following result.

Theorem 8 *Given the model class $\mathcal{M}_\theta$ (6) and approximator class $\mathcal{C}_\phi$ (7), the latter with linear activation functions, i.e., $\sigma = I$, then $f_{\mathcal{C}}(k)$ is an affine map of $\tilde{r}(k)$ according to*

$$f_{\mathcal{C}}(k) = \prod_{i=0}^{L} W_i \tilde{r}(k) + \sum_{i=0}^{L-1} b_i \prod_{l=i+1}^{L} W_l = W\tilde{r}(k) + b, \quad (12)$$

such that the least-squares cost J_{LS} (9) is given by

$$\sum_{j=1}^{N_{\mathcal{D}}} \sum_{k=1}^{N_j} \left| \hat{f}_j(k) - \left((\theta^T + W)\tilde{r}(k) + b \right) \right|^2. \quad (13)$$

Theorem 8 indicates that only the sum $\theta^T + W$ can be uniquely determined, i.e., $\mathcal{F}_{\theta,\phi}$ is not identifiable for $\mathcal{C}_\phi$ with linear activation functions. Hence, the optimum $\theta^*, \phi^* = \arg\min_{\theta,\phi} J_{LS}$ is not unique in θ^* for any dataset $\mathcal{D}$.

B. Case 2: Persistence of Excitation

A second source of non-uniqueness of θ^* is a dataset $\mathcal{D}$ that is not informative enough to distinguish between different coefficients θ, ϕ in $\mathcal{F}_{\theta,\phi}$, as formalized next [8].

Definition 9 *A dataset $\mathcal{D}$ is persistently exciting with respect to the identifiable parametrization $\mathcal{F}_{\theta,\phi}$ if, for any two realizations $\mathcal{F}_{\theta_1,\phi_1}$, $\mathcal{F}_{\theta_2,\phi_2}$,*

$$\sum_{j=1}^{N_{\mathcal{D}}} \sum_{k=1}^{N_j} \left(\mathcal{F}_{\theta_1,\phi_1}(r_j(k)) - \mathcal{F}_{\theta_2,\phi_2}(r_j(k)) \right)^2 = 0, \quad (14)$$

implies that $(\theta_1, \phi_1) = (\theta_2, \phi_2)$.

When $\mathcal{F}_{\theta,\phi}$ consists of just $\mathcal{M}_\theta$ in (6), conditions on $\mathcal{D}$ to uniquely identify θ are well-known in terms of the spectrum of r [8]. However, these results no longer apply to $\mathcal{M}_\theta$ when placed in parallel with $\mathcal{C}_\phi$ in $\mathcal{F}_{\theta,\phi}$, as illustrated next.

Example 10 *Consider $\mathcal{M}_\theta$ (6) of order $N_\theta = 2$, and $\mathcal{C}_\phi$ (7) with one hidden layer, i.e., $L = 1$, with ReLU activation functions, such that $\mathcal{F}_{\theta,\phi}$ is identifiable and given by*

$$f(k) = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}^T \begin{bmatrix} r(k) \\ r^{(1)}(k) \end{bmatrix} + W_1 \max\left(W_0 \begin{bmatrix} r(k) \\ r^{(1)}(k) \end{bmatrix} + b_0 \right).$$

Consider now the dataset $\mathcal{D}$ consisting of a single reference r for which $r(k), r^{(1)}(k) > 0\ \forall k$, and corresponding optimal input $\hat{f}(k) = c_0 r(k)$, $c_0 \in \mathbb{R}$. Note that r would be persistently exciting for just $\mathcal{M}_\theta$ [8]. However, the parameters $\theta_0 = c_0 + c_1$, $\theta_1 = 0$, $W_1 = [-c_1,\ 0]$, $W_0 = I$ and $b_0 = 0$ lead to $J_{LS} = 0$ for all values of $c_1 \in \mathbb{R}$, as

$$\begin{aligned} f(k) &= (c_0 + c_1)r(k) + \begin{bmatrix} -c_1 & 0 \end{bmatrix} \max\left(\begin{bmatrix} r(k) \\ r^{(1)}(k) \end{bmatrix} \right) \\ &= (c_0 + c_1)r(k) - c_1 r(k) = c_0 r(k) = \hat{f}(k). \end{aligned}$$

Thus the optimum $\theta^*, \phi^* = \arg\min_{\theta,\phi} J_{LS}$ is not unique.

Similar examples can be given for networks with more layers and different activation functions, illustrating that persistence of excitation conditions on $\mathcal{D}$ associated with linear models $\mathcal{M}_\theta$ are insufficient for a unique estimate θ^*.

C. Consequences of Non-Uniqueness

The non-uniqueness of θ^* directly results in uninterpretable models. Yet, it does not necessarily decrease performance as long as the realized IO map of $\mathcal{F}_{\theta,\phi}$ is equivalent. This can be the case inside the training regime. However, $\mathcal{D}$ usually does not cover the full space $\tilde{r}(k) \in \mathbb{R}^{N_\theta}$, and neural networks extrapolate poorly [17], such that $\mathcal{F}_{\theta,\phi}$ does not describe $\mathcal{J}$ outside the training regime. Thus, the non-uniqueness of θ deteriorates the task flexibility of $\mathcal{F}_{\theta,\phi}$ for references outside the training regime, which is not the case for just $\mathcal{M}_\theta$. As a special illustrative case, consider that g_y is zero outside a closed subset $\mathbb{X}$, i.e., $g_y(\tilde{y}(k)) = 0\quad \forall \tilde{y}(k) \notin \mathbb{X} \subset \mathbb{R}^{m+1}$. Then perfect extrapolation outside $\mathbb{X}$ is guaranteed for $a = \theta$ and $g_y = \mathcal{C}_\phi$. This cannot be realized if θ^* is not unique.

Non-uniqueness of θ^* through both unidentifiability and a lack of persistence of excitation are caused by the universal approximator characteristics of $\mathcal{C}_\phi$: the neural network is always able to generate an input map that is (locally) linear in $\tilde{r}(k)$. As a result, both $\mathcal{M}_\theta$ and $\mathcal{C}_\phi$ can capture the known dynamics, resulting in non-unique model coefficients θ.

More rigorous conditions could be imposed on $\mathcal{D}$ to guarantee unique coefficients θ. Instead, to avoid these complex conditions, uniqueness of θ^* is addressed by modifying the cost J_{LS} such that $\mathcal{M}_\theta$ is prioritized for fitting the known dynamics of $\mathcal{J}$. This modification imposes uniqueness of θ^* through the cost criterion, essentially favouring one realization of $\mathcal{F}_{\theta,\phi}$ over another based on physical insights.

IV. Uniqueness by Orthogonal Regularization

In this section, the orthogonal projection-based cost function is introduced, which decouples the optimization into orthogonal subspaces and regularizes the output of $\mathcal{C}_\phi$ in the subspace of $\mathcal{M}_\theta$. As a result, dynamics that can be described by $\mathcal{M}_\theta$, are captured by $\mathcal{M}_\theta$, resulting in interpretable coefficients θ. This constitutes the main contribution.

A. Model Output Space

An explicit basis of the output space of model component $\mathcal{M}_\theta$ for reference r_j can be derived by lifting the discrete time signal over the reference length N_j. Consider again the model parametrization (6). For the stacked reference

$$\underline{r}_j = \begin{bmatrix} r_j(1) & r_j(2) & \dots & r_j(N_j)\end{bmatrix}^T \in \mathbb{R}^{N_j}, \qquad (15)$$

the stacked response $\underline{f}_{\mathcal{M}}$ of $\mathcal{M}_\theta$ is given by

$$\underline{f}_{\mathcal{M},j} = \begin{bmatrix} \tilde{r}(1) & \dots \tilde{r}(N_j)\end{bmatrix}^T \theta = M(\underline{r}_j)\theta \in \mathbb{R}^{N_j}, \qquad (16)$$

with $M(\underline{r}_j) \in \mathbb{R}^{N_j \times N_\theta}$, $N_j > N_\theta$. The following is assumed.

Assumption 11 *The matrix representation $M(\underline{r}_j)$ of $\mathcal{M}_\theta$ in* (16) *has full rank N_θ for all references $r_j \in \mathcal{D}$.*

This corresponds to a persistently exciting dataset $\mathcal{D}$ with respect to the linear model class $\mathcal{M}_\theta$. An explicit basis for the output space of model $\mathcal{M}_\theta$ for reference r_j can be found by the singular value decomposition (SVD) of $M(\underline{r}_j)$.

Definition 12 *The singular value decomposition of full rank matrix $M(\underline{r}_j) \in \mathbb{R}^{N_j \times N_\theta}$, $N_j > N_\theta$ is the factorization*

$$M(\underline{r}_j) = \begin{bmatrix} U_1(\underline{r}_j) & U_2(\underline{r}_j)\end{bmatrix} \begin{bmatrix} \Sigma(\underline{r}_j) \\ 0 \end{bmatrix} V^T(\underline{r}_j), \qquad (17)$$

where $U_1(\underline{r}_j) \in \mathbb{R}^{N_j \times N_\theta}$, $U_2(\underline{r}_j) \in \mathbb{R}^{N_j \times N_j - N_\theta}$ satisfy

$$U_1^T(\underline{r}_j)U_1(\underline{r}_j) = I_{N_\theta},\ U_1^T(\underline{r}_j)U_2(\underline{r}_j) = 0, \qquad (18)$$

and similarly for $U_2(\underline{r}_j)$.

By Definition 12, the response of $\mathcal{M}_\theta$ can be written as

$$\underline{f}_{\mathcal{M},j} = M(\underline{r}_j)\theta = U_1(\underline{r}_j)\Sigma(\underline{r}_j)V^T(\underline{r}_j)\theta, \qquad (19)$$

in which the columns of $U_1(\underline{r}_j)$ form a basis for the output space of $\mathcal{M}_\theta$ for reference r_j, independent of parameters θ, due to linearity in the parameters of $\mathcal{M}_\theta$.

B. Orthogonal Regularization

To prevent $\mathcal{C}_\phi$ from learning an input map that could also be represented by $\mathcal{M}_\theta$, i.e., to prioritize the model for fitting the modelled dynamics of $\mathcal{J}$, the stacked output of $\mathcal{C}_\phi$ that lies in the output space of the model spanned by $U_1(\underline{r}_j)$ is penalized through regularization. The stacked output of $\mathcal{C}_\phi$ for $\underline{r}_j$ is denoted by $\underline{f}_{\mathcal{C},j} \in \mathbb{R}^{N_j}$ given by

$$\underline{f}_{\mathcal{C},j} = \begin{bmatrix} \mathcal{C}_\phi\left(\tilde{r}(1)\right) & \dots & \mathcal{C}_\phi\left(\tilde{r}(N_j)\right)\end{bmatrix}^T = \mathcal{C}_\phi(\underline{r}_j). \qquad (20)$$

The following lemma allows for expressing the component of $\underline{f}_{\mathcal{C},j}$ in the subspace spanned by $U_1(\underline{r}_j)$ [22].

Lemma 13 *For models $\mathcal{M}_\theta(r_j)$ that are linear in the parameters with finite-time response $U_1(\underline{r}_j)\Sigma(\underline{r}_j)V^T(\underline{r}_j)\theta$ to reference $\underline{r}_j$, the projection onto the subspace spanned by the columns of $U_1(\underline{r}_j)$ is given by*

$$\Pi_1(\underline{r}_j) = U_1(\underline{r}_j)U_1^T(\underline{r}_j), \qquad (21)$$

and the projection onto the orthogonal complement spanned by the columns of $U_2(\underline{r}_j)$ is given by

$$\Pi_2(\underline{r}_j) = U_2(\underline{r}_j)U_2^T(\underline{r}_j) = I - U_1(\underline{r}_j)U_1^T(\underline{r}_j). \qquad (22)$$

Thus, the component of $\underline{f}_{\mathcal{C},j}$ in the subspace spanned by $U_1(\underline{r}_j)$ is given by $\Pi_1(\underline{r}_j)\mathcal{C}_\phi(\underline{r}_j)$. Next, this component is used as regularization to obtain the orthogonal projection-based cost function, constituting the main contribution.

Definition 14 *The cost function $J_P \in \mathbb{R}_{\geq 0}$ is defined as the regularized least-squares cost according to*

$$J_P = \sum_{j=1}^{N_\mathcal{D}} \left(\|\underline{\hat{f}}_j - \left(M(\underline{r}_j)\theta + \mathcal{C}_\phi(\underline{r}_j)\right)\|_2^2 + \lambda R(\underline{r}_j)\right), \qquad (23)$$

with $\lambda \in \mathbb{R}_{\geq 0}$ the regularization weight, and $R(\underline{r}_j) : \mathbb{R}^{N_j} \to \mathbb{R}_{\geq 0}$ the orthogonality-promoting regularization given by

$$R(\underline{r}_j) = \|\Pi_1(\underline{r}_j)\mathcal{C}_\phi(\underline{r}_j)\|_2^2. \qquad (24)$$

Remark 15 *The regularization $R(\underline{r}_j)$ can be interpreted as targeted L_2 regularization that only shrinks directions of ϕ that generate output in the subspace of the model $\mathcal{M}_\theta$. As components of $\underline{\hat{f}}$ in this subspace can be captured by $\mathcal{M}_\theta$ by construction, $R(\underline{r}_j)$ does not reduce performance.*

This regularization promotes orthogonality between $\underline{f}_{\mathcal{C},j}$ and *any* $\underline{f}_{\mathcal{M},j}$ through penalizing the component of $\underline{f}_{\mathcal{C},j}$ in the subspace spanned by $U_1(\underline{r}_j)$, due to the linearity in the parameters of $\mathcal{M}_\theta$ (6). As a result, modelled effects are penalized from being included in the approximator, as formalized in the following theorem.

Theorem 16 *Given the model class $\mathcal{M}_\theta$* (6), *approximator class $\mathcal{C}_\phi$* (7) *and cost function J_P* (23), *the optimization*

$$\theta^*, \phi^* = \arg\min J_P, \qquad (25)$$

can be equivalently written as

$$\arg\min_{\theta,\phi} J_P = \arg\min_{\theta,\phi} J_1(\theta,\phi) + J_2(\theta,\phi) + J_3(\phi), \qquad (26)$$

in which $J_i \in \mathbb{R}_{\geq 0}$ are given by

$$J_1(\theta,\phi) = \sum\nolimits_{j=1}^{N_\mathcal{D}} \|\Pi_1(\underline{r}_j)\underline{\hat{f}}_j - U_1(\underline{r}_j)\Sigma(\underline{r}_j)V^T(\underline{r}_j)\theta - \Pi_1(\underline{r}_j)\mathcal{C}_\phi(\underline{r}_j)\|_2^2, \qquad (27)$$

$$J_2(\theta,\phi) = \sum\nolimits_{j=1}^{N_\mathcal{D}} \|\Pi_2(\underline{r}_j)\left(\underline{\hat{f}}_j - \mathcal{C}_\phi(\underline{r}_j)\right)\|_2^2, \qquad (28)$$

$$J_3(\phi) = \lambda \sum\nolimits_{j=1}^{N_\mathcal{D}} \|\Pi_1(\underline{r}_j)\mathcal{C}_\phi(\underline{r}_j)\|_2^2. \qquad (29)$$

Corollary 17 *If $\mathcal{J}$ is linear, i.e., $g_y(\tilde{y}) = 0\ \forall \tilde{y}$, and the model order is greater than or equal to the system order, i.e., $N_\theta \geq m+1$, then $\underline{\hat{f}}_j \in im\ U_1(\underline{r}_j)$, and $\mathcal{C}_\phi = 0$.*

These contributions create a trade-off between i) capturing g_y with $\mathcal{C}_\phi$ to increase performance (J_2) and ii) orthogonality of $\mathcal{M}_\theta$ and $\mathcal{C}_\phi$ (J_3). As a result, the modelled dynamics of $\mathcal{J}$ are primarily captured by $\mathcal{M}_\theta$ (J_1). Note that ϕ^* remains non-unique by the structure of universal approximator (7).

Remark 18 *The cost function J_{LS} could recover the same optimum θ^*, ϕ^* as J_P. However, the regularization* (24) *explicitly shapes the cost landscape to recover the orthogonal projection-based solution, whereas for J_{LS} this depends on the initialization of ϕ, see Section V.*

The orthogonal projection-based cost function J_P gives direct control over the measure of orthogonality between $\mathcal{M}_\theta$ and $\mathcal{C}_\phi$ through hyperparameter λ, and can be used to ensure that the modelled dynamics of $\mathcal{J}$ are explained by $\mathcal{M}_\theta$.

V. Simulation Example

In this section, a feedforward filter $\mathcal{F}_{\theta,\phi}$ with cost function J_P is demonstrated on an example dynamic system according to Definition 1, showing that it outperforms both an approach based solely on $\mathcal{M}_\theta$, as well as an approach based on a parallel combination $\mathcal{F}_{\theta,\phi}$ with criterion J_{LS}.

A. Example System

The dynamic process $\mathcal{J}$ is given by a mass-damper system with Stribeck-like friction characteristics often found in positioning systems with linear guidance and ball bearings, e.g., stage systems for lithographic inspection tools, i.e.,

$$my^{(2)}(k) + c_1 y^{(1)}(k) + \frac{c_2 - c_1}{\cosh\left(\alpha y^{(1)}(k)\right)} y^{(1)}(k) = f(k), \tag{30}$$

with parameters $c_1 = 1$, $c_2 = 20$, $\alpha = 20$ and $m = 5$, i.e., $a = [0,\ 1,\ 5]^T$. The friction characteristics are visualized in Fig. 3, in which the nonlinear contribution is negligible outside $\mathbb{X} = \mathbb{R} \times [-0.6, 0.6] \times \mathbb{R}$. For the system (30), a dataset of $N_\mathcal{D} = 9$ fourth-order references is generated combined with the optimal input $\hat{f}$ for each reference.

B. Simulation Results

In this section, three feedforward parametrizations and associated optimization criteria are compared. These are

1) A linear model $\mathcal{M}_\theta$ (6) optimized with J_{LS} (9).
2) A parallel filter $\mathcal{F}_{\theta,\phi}$ (5) optimized with J_{LS} (9).
3) $\mathcal{F}_{\theta,\phi}$ (5) optimized with J_P (23) with $\lambda = 0.01$.

In all cases, the model $\mathcal{M}_\theta$ is parametrized by a second-order system, i.e., $N_\theta = 3$, according to

$$\mathcal{M}_\theta : f_\mathcal{M}(k) = \theta_0 r(k) + \theta_1 r^{(1)}(k) + \theta_2 r^{(2)}(k). \tag{31}$$

In parametrization 2) and 3), $\mathcal{C}_\phi$ is parametrized as a neural network with $L = 2$ hidden layers with five neurons each and $\tanh$ activation functions, three input neurons and one linear output neuron, i.e.,

$$\mathcal{C}_\phi : f_\mathcal{C}(k) = W_2 \tanh(W_1 \tanh(W_0 \tilde{r}(k) + b_0) + b_1), \tag{32}$$

with $\phi = \{W_0, W_1, W_2, b_0, b_1\}$ the parameters of the network, $W_0 \in \mathbb{R}^{5\times 3}$, $W_1 \in \mathbb{R}^{5\times 5}$, $W_2 \in \mathbb{R}^{1\times 5}$, and $\tilde{r}^T(k) = \begin{bmatrix} r(k) & r^{(1)}(k) & r^{(2)}(k) \end{bmatrix}$.

Each parametrization is optimized according to the associated criterion with 50 LBFGS iterations, followed by ADAM until convergence. Fig. 4 shows the generated feedforward signal f by $\mathcal{M}_\theta$ (Fig. 4a), $\mathcal{F}_{\theta,\phi}$ with cost function J_{LS} (Fig. 4b) and $\mathcal{F}_{\theta,\phi}$ with J_P (Fig. 4c), for a reference r similar to $\mathcal{D}$ in terms of position, velocity and acceleration (top), and one dissimilar to $\mathcal{D}$ (bottom) with higher accelerations and velocities, requiring extrapolation outside the training dataset. The following observations are made.

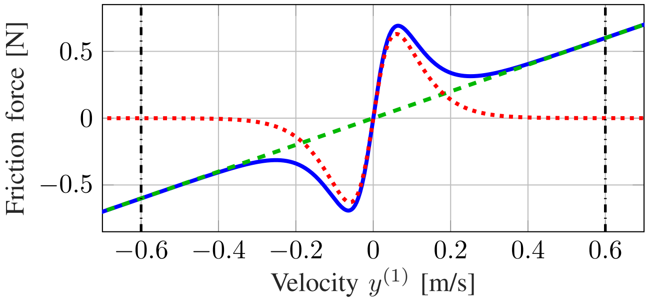


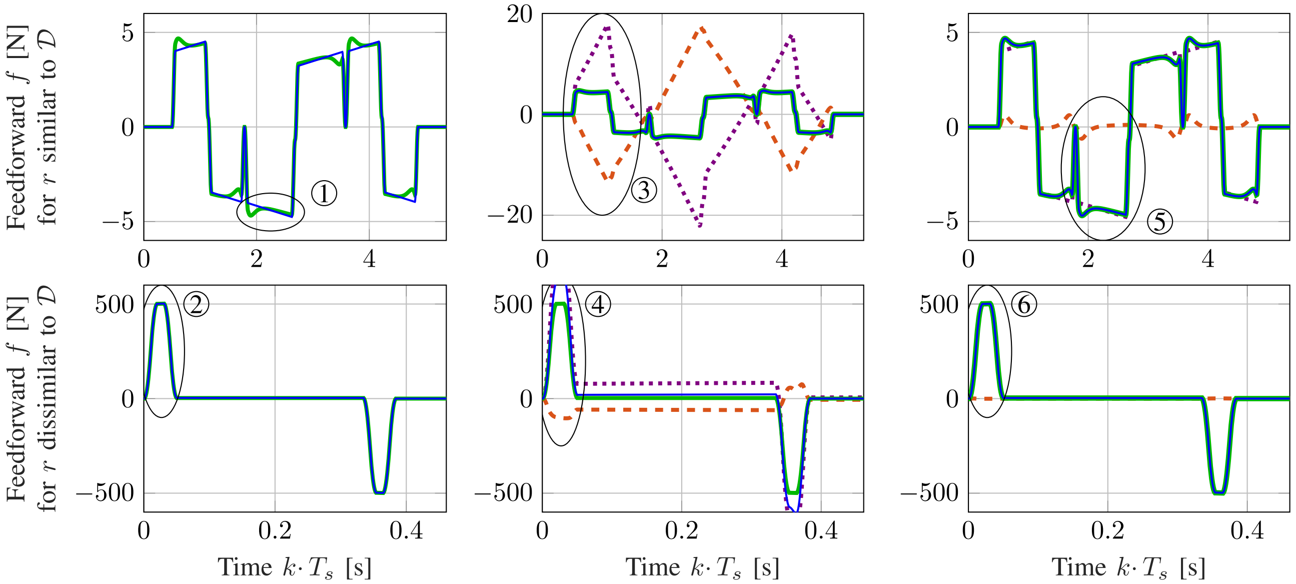
Fig. 3: Stribeck-like friction curve of example system (30) with $c_1 = 1$, $c_2 = 20$, $\alpha = 20$. The nonlinearity g_y (⋯) is approximately zero outside $\mathbb{X}$ (-·-). The full friction curve (—) approximately coincides with the linear viscous damping (- -) for high velocities.

- $\mathcal{M}_\theta$ (Fig. 4a) is not able to capture the unmodelled dynamics in $\hat{f}$: the linear model class limits performance (top). However, specifically due to this prior in the form of a linear model, $\mathcal{M}_\theta$ has satisfactory task flexibility (bottom) with parameters $\theta^* = [0.0,\ 1.18,\ 5.00]^T$.
- $\mathcal{F}_{\theta,\phi}$ with cost function J_{LS} (Fig. 4b) is able to capture the unmodelled dynamics in $\hat{f}$ and thus has good performance for references in and similar to $\mathcal{D}$ (top). However, due to non-uniqueness, the model with parameters $\theta^* = [6.30,\ 25.91,\ 6.90]^T$ is physically inconsistent (no stiffness is present in $\mathcal{J}$), and the parameters do not equal the true parameters of the linear dynamics a resulting in bad task flexibility outside the training set (bottom).
- $\mathcal{F}_{\theta,\phi}$ with cost function J_P (Fig. 4c) is able to capture the unmodelled dynamics in $\hat{f}$ with $\mathcal{C}_\phi$ resulting in good performance for references in and similar to $\mathcal{D}$ (top). Additionally, due to the orthogonal projection-based cost function, it has interpretable model coefficients $\theta^* = [0,\ 1.17,\ 5.00]$ that accurately describe the true dynamics of $\mathcal{J}$ with coefficients a, resulting in good task flexibility outside the training set (bottom), such that $f_\mathcal{M} \approx \hat{f}$ and $f_\mathcal{C} \approx 0$.

This simulation shows that $\mathcal{F}_{\theta,\phi}$ combined with J_P has both good performance and good task flexibility, as it is able to capture unmodelled nonlinear effects with $\mathcal{C}_\phi$, and to extrapolate based on the modelled dynamics $\mathcal{M}_\theta$.

VI. Conclusion

In this paper, a feedforward control framework is introduced that enables superior performance over model-based feedforward control, while maintaining task flexibility and interpretability. A physics-based model that is linear in its parameters is complemented by a neural network to obtain a parallel feedforward filter. For this parallel filter, optimization of its coefficients with a least-squares criterion results in



(a) Physics-based model $\mathcal{M}_\theta$ with J_{LS}. (b) Parallel combination $\mathcal{F}_{\theta,\phi}$ with J_{LS}. (c) Parallel combination $\mathcal{F}_{\theta,\phi}$ with $J_{P,0.01}$.

Fig. 4: Optimal (—) and generated (—) feedforward with model component $f_{\mathcal{M}}$ (⋯) and approximator component $f_{\mathcal{C}}$ (- -). The model $\mathcal{M}_\theta$ is not able to capture the unmodelled nonlinearity ①, and thus has limited performance, but generalizes well due to its linear structure ②. The parallel combination $\mathcal{F}_{\theta,\phi}$ with cost function J_{LS} is able to capture the unmodelled nonlinear dynamics resulting in high performance for references similar to $\mathcal{D}$ in terms of velocity and acceleration ③, but has opposing contributions $f_{\mathcal{M}} \approx -f_{\mathcal{C}}$ due to non-uniqueness of θ^*, and therefore fails to extrapolate ④. $\mathcal{F}_{\theta,\phi}$ with orthogonal projection-based cost function J_P is both able to capture unmodelled nonlinear dynamics ⑤, and to extrapolate based on the modelled dynamics ⑥, resulting in high performance, task flexibility, and interpretability.

non-unique model coefficients, limiting interpretability and task flexibility. To address this non-uniqueness, an orthogonal projection-based cost function is derived that penalizes the output of the neural network in the subspace of the model through regularization, such that an interpretable model is obtained that accurately captures the modelled part of the system dynamics, enabling task flexibility outside the training dataset. Future work focuses on guaranteeing that the approximator is also regularized to a desired output outside the training dataset, e.g., zero, and extending the approach to flexible dynamics with rational model structures.

References

[1] J. A. Butterworth, L. Y. Pao, and D. Y. Abramovitch, “A comparison of control architectures for atomic force microscopes,” *Asian J. Control*, vol. 11 (2), pp. 175–181, 2009.

[2] Q. Zou and S. Devasia, “Preview-based optimal inversion for output tracking: Application to scanning tunneling microscopy,” *IEEE Trans. Control Syst. Technol.*, vol. 12 (3), pp. 375–386, 2004.

[3] P. Lambrechts, M. Boerlage, and M. Steinbuch, “Trajectory planning and feedforward design for electromechanical motion systems,” *Control Eng. Pract.*, vol. 13 (2), pp. 145–157, 2005.

[4] M. Boerlage, M. Steinbuch, P. Lambrechts, and M. Van De Wal, “Model-based feedforward for motion systems,” *IEEE Conf. Control Appl. - Proc.*, vol. 2, pp. 1158–1163, 2003.

[5] S. Devasia, D. Chen, and B. Paden, “Nonlinear inversion-based output tracking,” *IEEE Trans. Automat. Contr.*, vol. 41 (7), pp. 930–942, 1996.

[6] L. R. Hunt, G. Meyer, and R. Su, “Noncausal inverses for linear systems,” *IEEE Trans. Automat. Contr.*, vol. 41 (4), pp. 608–611, 1996.

[7] J. Bolder and T. Oomen, “Rational basis functions in iterative learning control - With experimental verification on a motion system,” *IEEE Trans. Control Syst. Technol.*, vol. 23 (2), pp. 722–729, 2015.

[8] L. Ljung, *System Identification: Theory for the User*, 2nd ed., T. Kailath, Ed. Prentice Hall PTR, 1999.

[9] J. van Zundert and T. Oomen, “On inversion-based approaches for feedforward and ILC,” *Mechatronics*, vol. 50, pp. 282–291, 2018.

[10] M. Tomizuka, “Zero phase error tracking algorithm for digital control,” *J. Dyn. Syst. Meas. Control*, vol. 109 (1), pp. 65–68, 1987.

[11] S. Devasia, “Robust inversion-based feedforward controllers for output tracking under plant uncertainty,” *Proc. Am. Control Conf.*, vol. 1, pp. 497–502, 2000.

[12] O. Sørensen, “Additive feedforward control with neural networks,” *IFAC Proc. Vol.*, vol. 32 (2), pp. 1378–1383, 1999.

[13] K. J. Hunt, D. Sbarbaro, R. Zbikowski, and P. J. Gawthrop, “Neural networks for control systems—A survey,” *Automatica*, vol. 28 (6), pp. 1083–1112, 1992.

[14] G. Otten, T. J. De Vries, J. Van Amerongen, A. M. Rankers, and E. W. Gaal, “Linear motor motion control using a learning feedforward controller,” *IEEE/ASME Trans. Mechatronics*, vol. 2 (3), pp. 179–187, 1997.

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[16] L. Ljung, C. Andersson, K. Tiels, and T. B. Schön, “Deep learning and system identification,” *IFAC-PapersOnLine*, vol. 53 (2), pp. 1175–1181, 2020.

[17] K. Xu, M. Zhang, J. Li, S. S. Du, K.-I. Kawarabayashi, and S. Jegelka, “How neural networks extrapolate: From feedforward to graph neural networks,” *Int. Conf. Learn. Represent.*, vol. 9, 2020.

[18] A. Karpatne, G. Atluri, J. H. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar, “Theory-guided data science: A new paradigm for scientific discovery from data,” *IEEE Trans. Knowl. Data Eng.*, vol. 29 (10), pp. 2318–2331, 2017.

[19] A. Karpatne, W. Watkins, J. Read, and V. Kumar, “Physics-guided neural networks (PGNN): An application in lake temperature modeling,” *arXiv*, 2017.

[20] M. Bolderman, M. Lazar, and H. Butler, “Physics-guided neural networks for inversion-based feedforward control applied to linear motors,” *arXiv*, 2021.

[21] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 770–778, 2016.

[22] D. A. Freedman, *Statistical Models: Theory and Practice*, 2nd ed. Cambridge University Press, 2009.

[23] J. N. Hendriks, C. Jidling, A. Wills, and T. Schön, “Linearly Constrained Neural Networks,” *arXiv*, 2020.