Nonlinear Model Reference Adaptive Control (NMRAC) for First Order Systems

Dalim Wahby
Université Côte d'Azur
I3S/CNRS
Sophia Antipolis, France
wahby@i3s.unice.fr

Alvaro Detailleur
IDSC
ETH Zürich
Zürich, Switzerland
adetailleur@student.ethz.ch

Guillaume Ducard
Université Côte d'Azur
I3S/CNRS
Sophia Antipolis, France
ducard@i3s.unice.fr

***Abstract*—...**

***Keywords*—Neural Networks, Gradient Descent, Lyapunov Method, Nonlinear Control, Stability, Adaptive Control, Robotics**

I. Introduction

...

II. Literature Review and Contributions

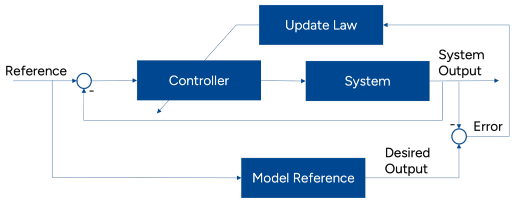


Figure 1: MRAC for first-oder systems controlled by simple NNCs

A. Contributions

III. Definitions and Nomenclature

A. Neural Network

The controllers developed in this work are considered to be neural networks (NN).

Definition III.1 (Neural network)**.** A neural network (NN) $\phi : \mathbb{R}^n \to \mathbb{R}^p$ is defined as:

$$\begin{aligned} \phi(x) &= (L_1 \circ \varphi_1 \cdots \circ L_{H-1} \circ \varphi_{H-1} \circ L_H \circ \varphi_H)(x) \\ L_i(x) &= \theta_i x + b_i \quad \forall i \in \{1, ..., H\}, \end{aligned} \tag{1}$$

where $\varphi_i(\cdot)$ are called activation functions, θ_i and b_i are the weight matrix and bias of layer i, respectively.

B. Stability in the sense of Lyapunov

The proposed method is based on the concept of stability in the sense of Lyapunov. The used definitions for stability are taken from [1].

Definition III.2 (Lyapunov function)**.**

Definition III.3 (Stability in the sense of Lyapunov)**.** If there exists a continuous function $V : \mathbb{R}^n \to \mathbb{R}$ such that

(a) $V(x)$ is positive definite, and
(b) $\dot{V}(x) \leq -l(x)$, for some positive semidefinite function $l(x)$,

then the system is considered to be stable. Additionally, if $l(x)$ is positive definite, then the system is asymptotically stable. The function V is called a Lyapunov function.

C. Smooth saturation function

Throughout this work, a saturation function is used as an activation function for the neural network. Typically, a tangens hyperbolicus function is used in machine learning to model saturation, however, in this work a sigmoid based, as defined in (2), saturation function is applied, as done in **wahby, thanh**. Note, that this activation function saturates at $\pm\frac{2}{a}$.

$$\sigma(x) = \frac{2(1 - e^{-ax})}{a(1 + e^{-ax})} \tag{2}$$

IV. Nonlinear Model Reference Adaptive Control (NMRAC)

One of the goal of this work is to analyse stability properties of nonlinear NNCs. To this end, we propose stable update scheme. We commence by looking at the simpler case of a first-order system. More, specifically, we define a problem, where we show that the even in the nonlinear case, we are able to learn a parameter with an algorithm that is based on the logic of a linear MRAC. We call this nonlinear Model Reference Adaptive Control (NMRAC).

We desire to control a system of the form, as described in equation (3), where $\phi : \mathbb{R} \to \mathbb{R}$ is a nonlinear function, $e_x = x_r - x$,and θ is the parameter we aim to learn. Here, we would like to point that θ can be seen as a weight, and ϕ the activation function of a NN of the form of a sigmoidial saturation function, as described in equation (2).

$$\dot{x} = -ax + b\phi(\theta e_x) \tag{3}$$

Next, we define a stable reference model, as defined in equation (4), where $e_m = x_r - x_m$, and $a_m > 0$. Note, that we want ϕ_m to be of the same form as ϕ.

$$\dot{x}_m = -a_m x_m + b_m \phi_m(\theta_m e_m) \tag{4}$$

Since we aim for the system to follow the reference model, we define the error dynamics as in equation (5). Note, that from this definition it follows that $e = e_x - e_m$. We will use this alternative definition later for the stability analysis of the update law.

$$e = x_m - x \tag{5}$$

Following the error dynamics, we define the Lyapunov candidate to be defined as in equation (6). The factor $c > 0$, is used to accelerate or decelerate the learning process. The norm of the error captures the distance between the internal states of the reference model and the system, and the term $||\alpha||_2^2$ captures the distance between the NNC and the desired NNC, which can be seen as the difference in dynamics between the controlled system and the model reference. Note, that both e and α should be 0, when our system follows the reference model and the desired parameters are learned. Futhermore, this property renders our Lyapunov candidate to be positive definite.

$$V(e, \alpha) = ||e||_2^2 + ||c\alpha||_2^2 \tag{6}$$

To ensure that the controlled system is stable, we require the time derivative of the Lyapunov function to be negative. Therefore, we analyze the behavior of the resulting time derivative of the Lyapunov function. The resulting equation is defined in (7).

$$\dot{V}(e, \alpha) = 2e\dot{e} + 2\alpha\dot{\alpha}c \tag{7}$$

We now construct the time derivative of the error dynamics, which is defined by equation (8). We extend the equation by $\pm(a_m x b_m \phi_m(\theta_m e_x))$, to construct the term α, which is now dependent on e_x, and θ. We are then left with two other terms, namely, $-a_m e$, and $\gamma_m(e_m, e_x)$.

$$\begin{aligned}
\dot{e} =& \dot{x}_m - \dot{x} \\
=& - a_m x_m + b_m \phi_m(\theta_m e_m) - (-ax + b\phi(\theta e_x)) \\
& \pm (a_m x + b_m \phi_m(\theta_m e_x)) \\
=& - a_m \underbrace{(x_m - x)}_{=e} \\
& + \underbrace{(a - a_m)x + b_m \phi_m(\theta_m e_x) - b\phi(\theta e_x)}_{=\alpha(e_x, \theta)} \\
& + \underbrace{b_m(\phi_m(\theta_m e_m) - \phi_m(\theta_m e_x))}_{=\gamma_m(e_m, e_x)}
\end{aligned} \tag{8}$$

$$\begin{aligned}
\dot{V}(e, \alpha) =& 2e(-a_m e + \alpha(e_x, \theta) + \gamma_m(e_m, e_x)) + 2\alpha c\dot{\alpha}(e_x, \theta) \\
=& - 2a_m e^2 \\
& + 2e\gamma_m(e_m, e_x) \\
& + 2\alpha(e_x, \theta)(e + c\dot{\alpha}(e_x, \theta))
\end{aligned} \tag{9}$$

By substituting equation (8) into equation (7), we get equation (9).

Note, that the first term $-2a_m e^2$ is always negative, since $a_m > 0$, and $e^2 > 0$. The second term will always be negative, due to the property of $e = e_x - e_m$. It follows that if $e < 0 \Rightarrow \gamma_m(e_m, e_x) > 0$, which implies that the second term is negative, and if $e > 0 \Rightarrow \gamma_m(e_m, e_x) < 0$, which again implies that the second term is negative. Hence, the second term reamins negative.

Since, we can already ensure that the first two terms of equation (9) are negative, it remains to show that the last term is always negative or equal to zero at all times. Since θ is dependent on time, this implies $\dot{\alpha}(e_x, \theta)$ will have a term that includes $\dot{\theta}$. Therefore, we choose $\dot{\theta}$, such that the third term in equation (9) is nullified. Note, that due to the way we construct the update law, stability is guaranteed, since the Lyapunov function is positive definite and simultaneously the decrease condition is satisfied.

$$0 = 2\alpha(e_x, \theta)(e + \dot{\alpha}(e_x, \theta)) \tag{10}$$

$$\begin{aligned}
\dot{\theta} =& \frac{1}{e_x b \frac{\partial \phi(\tilde{x})}{\partial \tilde{x}}|_{\tilde{x}=\theta e_x}} \\
& (\frac{1}{c} e + (a - a_m)\dot{x} + b_m \frac{\partial \phi_m(\tilde{x})}{\partial \tilde{x}}|_{\tilde{x}=\theta_m e_x} \theta_m \dot{e}_x) \\
& - \frac{\theta}{e_x} \dot{e}_x
\end{aligned} \tag{11}$$

When taking a closer look at the update rule in equation (11), it follows that we require to implement an update hold, when $e_x = 0$. This is reasonable, since whenever the state of our reference system is equal to the reference signal, we will not be able to extract information on how to change the weights in order for our system to converge towards the reference signal. In practice this means that we implement a threshold ε, where we do not update the weights.

V. Results

In this section, we will present our simulation for the proposed MRAC strategy for nonlinear NNCs. We carefully construct an example to show that with the proposed schematic we are able to learn the desired parameter. To this end, we choose to use a simple NNC that consists of only one neuron with an activation function, which can be seen as a smooth saturation function, as defined in (2). The parameters of the neural network consist of one value that is updated.

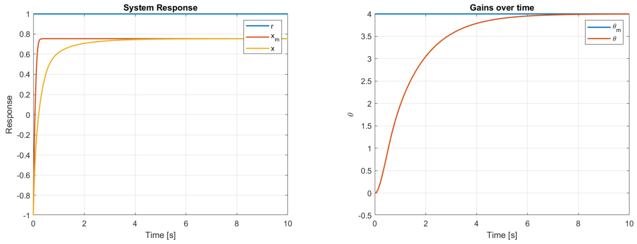
For simplicity, we choose the dynamics of the reference model and the system to be equal, and we choose the activation function to be the same. The dynamics of the system and the reference model have a form as defined in equation (3) and equation (4). The only parameters that are different are θ and θ_m. Hence, to ensure that the behavior of the reference model and the system are the same, we require θ to converge towards θ_m. We choose $a_m = a = 10$, $b_m = b = 10$, $\theta_m = 4$, and the activation function saturates at a value of 1. The precision threshold ϵ is chosen to be computer precision in all our simulations. We compute the weight update as the analytical solution for each time instance. The sampling time for our simulations is $T_s = 0.01$, and the learning rate differs in each simulation to showcase faster and slower convergence.

We will analyze the behavior with respect to different reference signals. Note, that we require the reference signal to be differentiable, because $\dot{r}$ is used in the update law. Hence, we will firstly look at the response of a constant signal, secondly a sine signal, and thirdly a smooth pulse signal.

A. Constant Reference Signal

We commence by using a constant reference signal. We choose the system's initial conditions to be at 0. The value of the constant reference signal is 1. This will result in a steady state error for the reference model since we only take a feedback controller into account. However, we will ignore this steady state error, since for us the ability of the controller to learn the behavior of the reference model is of interest. Additionally, we set $\theta_m = 4$ and the initial value of the weight $\theta = 0$. Finally, we simulate the system response for 5 seconds and choose the constant parameter $c = 0.05$.

In Figure 2a, we observe the steady state error, which is at around 0.2. As aforementioned, we will disregard this error, since we are interested in how well our NNC can learn the behavior of our reference model. To this end, we can observe that the system converges to the reference model within 5 seconds. Furthermore, in Figure 2b, we can see the weights of the NNC and the error e are equal to their desired values, $\theta = 4$ and $e = 0$ respectively. In Figure 3a, we can see that the Lyapunov function converges to zero over time. Additionally, we can see that the derivate of the Lyapunov function is negative and only reaches zero when the Lyapunov function itself converges to zero. This indicates that the proposed learning schematic is stable for the desired reference signal, and we can learn the desired parameter.

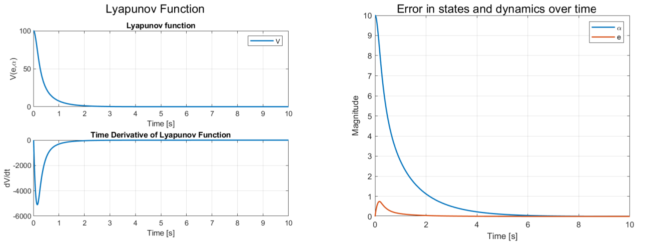


(a) System response (b) Evolution of gain update

Figure 2: Simulation results of nonlinear MRAC with a constant input

B. Sine Signal

The second reference is a sinusoid signal, with an amplitude of 0.25, to stay in a region that is not affected too much by the saturation function. As before, we choose the initial $\theta = 0$, $\theta_m = 4$, and the initial conditions of both the system and the reference model to be at -1. We choose this specific initial condition only to be able to see the convergence in the plots. Finally, we simulate the system response for 30 seconds and choose the constant parameter $c = 0.15$, which corresponds to a less aggressive learning strategy compared to the constant input signal.

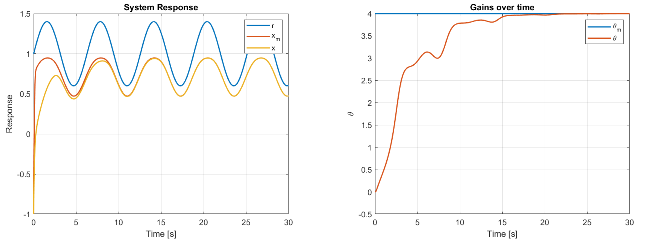


(a) Lyapunov function and its time derivate (b) Error in internal states and dynamics of the system

Figure 3: Lyapunov function and convergence of the system w.r.t. internal states and dynamics

Figure 4a shows the response of the system. We observe that the system state, and reference model state both converge towards the desired reference model within around 25 seconds. Simultaneously, the parameter θ converges to towards the desired value, which can be seen in Figure 4b. Even though the parameter converges, we observe oscillations. This behavior is expected, due to the update law. Since the update law is dependent on $\dot{e}_x$, it is dependent on $\dot{r}$ and $\dot{x}$. We observe that whenever $\dot{e}_x$ switches signs and simultaneously e is small, $\dot{\theta}$ changes sign, and hence the parameter starts oscillating. This behavior can be changed thourgh adapting the learning rate. A more aggressive learning strategy will be showcased in the next example.

In Figure 5a we can see that the Lyapunov function converges to zero over time. Additionally, we can see that the derivate of the Lyapunov function is negative and only reaches zero when the Lyapunov function itself converges to zero. The results of this simulation indicate again that the proposed learning schematic is stable for the desired reference signal, and we can learn the desired parameter.

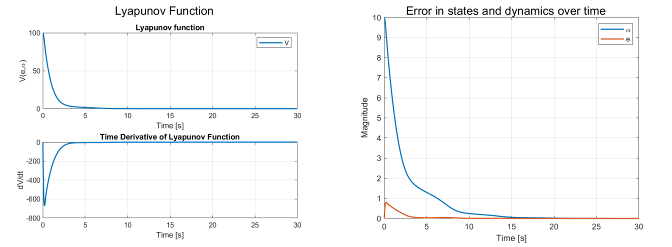


(a) System response (b) Evolution of gain update

Figure 4: Simulation results of nonlinear MRAC with a sine input

C. Smooth Pulse Signal

The third and last input signal we will look at is a pulse signal. More specifically, we will use a smooth pulse signal,



(a) Lyapunov function and its time derivate

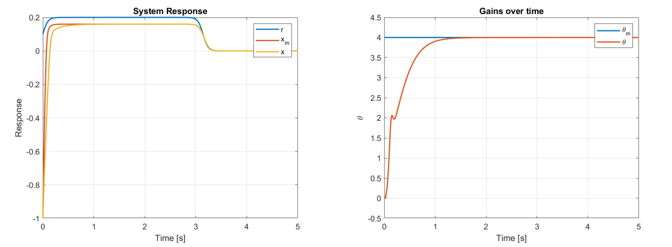
(b) Error in internal states and dynamics of the system

Figure 5: Lyapunov function and convergence of the system w.r.t. internal states and dynamics

because we require the reference signal to be continuous since we require its derivative in the update law. We construct the smooth pulse signal, by using a sine function and applying a smooth saturation function, as defined in equation (2). We choose the signal to have an amplitude of 0.25 and shift the signal such that its minimum value touches upon 0. As before, we choose the initial conditions of the system and plant to be -1. We simulate the system response for 5 seconds and choose the constant parameter $c = 0.005$.

Figure 6a shows the response of the system. We can observe that the system converges towards the desired reference model within just under 1 second. The parameter θ converges to towards the desired value within 2 seconds, which can be seen in Figure 6b. Even though the parameter converges, we observe that it oscillates. This behavior is expected, due to the update law. Since the update law is dependent on $\dot{e}_x$, it is dependent on $\dot{r}$ and $\dot{x}$. We observe that whenever $\dot{e}_x$ switches signs and simultaneously e is small, $\dot{\theta}$ changes sign, and hence the parameter starts oscillating. Similarly, as in the case of the sine input, this behavior can be influenced by decreasing the value of c.

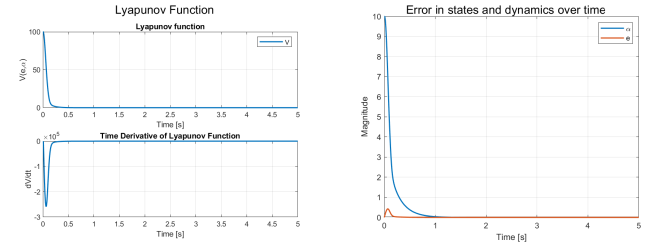
The Lyapunov function and its derivative, as shown in Figure 7a indicate the convergence of the system and the parameters, since V converges to 0 and its derivative is always negative and approach zero when V converges. The results of this simulation indicate that the proposed learning schematic is stable for the desired reference signal, and we are able to learn the desired parameter. The three simulation results strongly indicate that with the proposed update law we are able to learn the desired parameters in the presence of a nonlinear activation function.



(a) System response

(b) Evolution of gain update

Figure 6: Simulation results of nonlinear MRAC with a smooth pulse input



(a) Lyapunov function and its time derivate

(b) Error in internal states and dynamics of the system

Figure 7: Lyapunov function and convergence of the system w.r.t. internal states and dynamics

VI. Conclusion

The simulation results of the scalar system indicate that the proposed update law successfully enables the system to learn the desired parameters across different reference signals. The speed of convergence can be controlled by adjusting the factor c, with smaller values leading to faster convergence. Additionally, this work identifies numerical instability as a potential issue. This instability can be reduced by lowering the sampling time, thus enhancing the precision of integral and derivative approximations, improving overall simulation accuracy.

Acknowledgment

...

References

[1] H. K. Khalil, *Nonlinear Systems*, en. Prentice Hall, 1996, Google-Books-ID: qiBuQgAACAAJ, ISBN: 978-0-13-228024-2.