Performance Analysis of Model-Free Adaptive Control

Feilong Zhang

Abstract—We analyzed the model-free adaptive control (MFAC) through the closed-loop function to study its essence and the correct way of reposing it.

Index Terms—model-free adaptive control.

I. INTRODUCTION

A significant number of works on MFAC have been published in the past decade. We analyzed this method using a closed-loop function and obtained results that differ from those presented in three representative works on current MFAC [1]-[3].

i) The sign of the estimated leading coefficient of the control input is restricted to be unchangeable in the controller [1]-[3]. However, the leading coefficient in the actual model may change, causing the failure of the controller. Formula (13) in [1] lets \( \hat{\phi}_{\tilde{L}_1,\tilde{L}_n}(k) = \hat{\phi}_{\tilde{L}_1,\tilde{L}_n}(1) \) if \( \text{sign}(\hat{\phi}_{\tilde{L}_1,\tilde{L}_n}(k)) \neq \text{sign}(\hat{\phi}_{\tilde{L}_1,\tilde{L}_n}(1)) \), and this is the necessary precondition for the system stability analysis through the current contraction mapping technique [1]-[3]. Consequently, this restriction hinders the established model, namely the equivalent-dynamic-linearization model (EDLM), from accurately reflecting the actual system, potentially leading to controller failure. We have discussed this problem in Example 1.

ii) The static error of the speed response of the system is eliminated when \( \bar{\lambda} = 0 \), as proven in this brief. This conclusion differs from [1]-[3], which showed that the tracking error of the system controlled by MFAC converges to zero when \( \bar{\lambda} \) is large enough. Furthermore, when the desired trajectory is a step signal, the reason behind the convergence of the tracking error to zero in the MFAC-controlled system is the inherent presence of an integrator in the current MFAC rather than being determined by \( \bar{\lambda} \).

iii) The orders of function (1) in [1] should not be \( n_s, n_a \) but \( n_s+1, n_a+1 \) [6]. The two pseudo orders \( L_s \) and \( L_a \) in current works are limited to \( 1 \leq L_s \leq n_s \) and \( 1 \leq L_a \leq n_a \), respectively. However, according to [4]-[6], the most important and ideal choice of pseudo gradient orders is the actual gradient orders \( L_s = n_s+1 \) and \( L_a = n_a+1 \) in adaptive control. To this end, we extend the range to \( 0 \leq L_s \) and \( 1 \leq L_a \).

Furthermore, we have analyzed the MFAC for the multivariable systems in [7]-[9].

On the other hand, some noteworthy merits of the proposed method are shown as follows.

In comparison to previous works, our proposed equivalent-dynamic-linearization model (EDLM) offers the advantage of describing NARMAM at any point, making it easier and more widely applicable for MFAC. On the other hand, can anyone achieve the tracking performance of Example 3 within 15 minutes? After mastering the proposed method, one can easily achieve that.

The remainder of this paper is organized as follows: Section II presents the corrected EDLM and MFAC. Then the stability of the system controlled by MFAC is analyzed through the closed-loop system function and the simulations are presented. Section III studies how to apply MFAC in nonlinear systems. Section IV gives the conclusion.

II. EQUIVALENT DYNAMIC LINEARIZATION MODEL AND DESIGN OF MODEL-FREE ADAPTIVE CONTROL

In Part A of this section, we correct the current EDLM and reintroduce its fundamental assumptions and theorem. In Part B, we design the MFAC controller and analyze its performance.

A. Equivalent Dynamic Linearization Model

We consider the following discrete-time SISO system:

\[
y(k+1) = f(y(k),...,y(k-n_s),u(k),...,u(k-n_a)) \quad (1)
\]

where \( f(\cdot) \in \mathbb{R} \) represents the nonlinear function; \( u(k) \) and \( y(k) \) denote the input and output of the system at time \( k \), respectively. And \( n_s+1, n_a+1 \in \mathbb{Z} \) represent their orders.

Let

\[
\phi(k) = [y(k),...,y(k-n_s),u(k),...,u(k-n_a)]^T \quad (2)
\]

Then (1) can be rewritten as

\[
y(k+1) = f(\phi(k)) \quad (3)
\]

Assumption 1: The partial derivatives of \( f(\cdot) \) with respect to all variables are continuous.

Theorem 1: Given the nonlinear system (1) satisfying Assumptions 1, if \( \Delta H(k) \neq 0 \), there is a time-varying vector named PG vector, and the system (1) can be transformed into the following full-form equivalent-dynamic-linearization model

\[
\Delta y(k+1) = \phi^T(k) \Delta H(k) \quad (4)
\]

where

\[
\phi(k) = [\phi_1(k),\phi_2(k),\phi_3(k),...,\phi_{L_s+1}(k),...,\phi_{L_a+n_a}(k)]^T
\]

[10]-[14] decompose the nonlinear auto regressive and moving average model (NARMA) or nonlinear industrial process model into a simple linear model and an unmodeled dynamics (UD) around an operating point. Then the corresponding controller with compensation of UD is designed and analyzed using the closed-loop system equation. In comparison to previous works, our proposed equivalent-dynamic-linearization model (EDLM) offers the advantage of describing NARMAM at any point, making it easier and more widely applicable for MFAC. On the other hand, can anyone achieve the tracking performance of Example 3 within 15 minutes? After mastering the proposed method, one can easily achieve that.

The remainder of this paper is organized as follows: Section II presents the corrected EDLM and MFAC. Then the stability of the system controlled by MFAC is analyzed through the closed-loop system function and the simulations are presented. Section III studies how to apply MFAC in nonlinear systems. Section IV gives the conclusion.

B. Design of Model-Free Adaptive Controller

In Part A of this section, we correct the current EDLM and reintroduce its fundamental assumptions and theorem. In Part B, we design the MFAC controller and analyze its performance.

A. Equivalent Dynamic Linearization Model

We consider the following discrete-time SISO system:

\[
y(k+1) = f(y(k),...,y(k-n_s),u(k),...,u(k-n_a)) \quad (1)
\]

where \( f(\cdot) \in \mathbb{R} \) represents the nonlinear function; \( u(k) \) and \( y(k) \) denote the input and output of the system at time \( k \), respectively. And \( n_s+1, n_a+1 \in \mathbb{Z} \) represent their orders.

Let

\[
\phi(k) = [y(k),...,y(k-n_s),u(k),...,u(k-n_a)]^T \quad (2)
\]

Then (1) can be rewritten as

\[
y(k+1) = f(\phi(k)) \quad (3)
\]

Assumption 1: The partial derivatives of \( f(\cdot) \) with respect to all variables are continuous.

Theorem 1: Given the nonlinear system (1) satisfying Assumptions 1, if \( \Delta H(k) \neq 0 \), there is a time-varying vector named PG vector, and the system (1) can be transformed into the following full-form equivalent-dynamic-linearization model

\[
\Delta y(k+1) = \phi^T(k) \Delta H(k) \quad (4)
\]

where

\[
\phi(k) = [\phi_1(k),\phi_2(k),\phi_3(k),...,\phi_{L_s+1}(k),...,\phi_{L_a+n_a}(k)]^T
\]

[10]-[14] decompose the nonlinear auto regressive and moving average model (NARMA) or nonlinear industrial process model into a simple linear model and an unmodeled dynamics (UD) around an operating point. Then the corresponding controller with compensation of UD is designed and analyzed using the closed-loop system equation. In comparison to previous works, our proposed equivalent-dynamic-linearization model (EDLM) offers the advantage of describing NARMAM at any point, making it easier and more widely applicable for MFAC. On the other hand, can anyone achieve the tracking performance of Example 3 within 15 minutes? After mastering the proposed method, one can easily achieve that.

The remainder of this paper is organized as follows: Section II presents the corrected EDLM and MFAC. Then the stability of the system controlled by MFAC is analyzed through the closed-loop system function and the simulations are presented. Section III studies how to apply MFAC in nonlinear systems. Section IV gives the conclusion.
and \( \Delta \mathbf{H}(k) = \begin{bmatrix} \Delta Y_{i}(k) \\ \Delta \mathbf{U}_{i}(k) \end{bmatrix} = [\Delta y(k), \ldots, \Delta y(k - L_y + 1)] \), is a vector that consists of an increment of control input within the time window \([k-L_y+1,k]\) and an increment of system output within the time window \([k-L_y+1,k]\). Two integers \(0 \leq L_y, 1 \leq L_u\) are named pseudo orders of the system.

And we define \( \phi_{j,0}(z^{-1}) = \phi_0(k) + \cdots + \phi_{j-1}(k)z^{-(j-1)} \), \( \phi_{j,1}(z^{-1}) = \phi_{j,0}(z^{-1}) + \cdots + \phi_{j+1,0}(k)z^{-(j+1)} \), and \( z^{-1} \) is the backward-shift operator.

**Remark 1:** [1] and [2] give the proof of Theorem 1 in the case of \(1 \leq L_y \leq n_y, 1 \leq L_u \leq n_u\) and we further prove Theorem 1 for the orders \(0 \leq L_y, 1 \leq L_u\) in Appendix.

We prefer \( L_y = n_y + 1 \) and \( L_u = n_u + 1 \) in applications if \( n_y \) and \( n_u \) can be obtained. Otherwise, we usually choose the proper \( L_y, L_u \) that satisfy \( n_y + 1 \leq L_y, n_u + 1 \leq L_u \) in adaptive control. One reason is that the online estimated coefficients of redundant items \( \Delta y(k - n_y - 1), \ldots, \Delta y(k - L_y + 1) \) and \( \Delta u(k - n_u - 1), \ldots, \Delta u(k - L_u + 1) \) might be close to zero, and simultaneously the estimated coefficients of \( \Delta y(k), \ldots, \Delta y(k - n_y), \Delta u(k), \ldots, \Delta u(k - n_u) \) will be closer to the actual values compared to \(0 \leq L_y \leq n_y, 1 \leq L_u \leq n_u\).

**B. Design of Model Free Adaptive Control**

We can rewrite (4) into (5). 

\[
y(k+1) = y(k) + \Phi^*(k)\Delta \mathbf{H}(k) 
\]

(5)

The object is to design a controller that optimizes output tracking performance in the sense that:

\[
J = \left[ y^*(k+1) - y(k+1) \right]^2 = \min \text{ in } \text{num} 
\]

(6)

where \( y^*(k+1) \) is the desired system output signal.

Substitute Equation (5) into Equation (6) and solve the optimization condition \( \partial J / \partial \Delta u(k) = 0 \), then we have:

\[
\Delta u(k) = \frac{1}{\Phi_{2,1}(k)} \left[ y^*(k+1) - y(k) - \sum_{i=1}^{L_y} \phi_i(k) \Delta y(k - i + 1) \right. \\
- \left. \sum_{i=L_y+1}^{L_u+L_y+1} \phi_i(k) \Delta u(k + L_y - i + 1) \right] 
\]

(7)

Herein, we change the coefficient \( \frac{1}{\Phi_{2,1}(k)} \) into \( \lambda(k) + \Phi_{2,1}(k) \) to prevent the denominator from being zero and let \( \lambda(k) = \lambda \) for easier performance analysis. Then the controller will become (8).

\[
\Delta u(k) = \frac{\Phi_{2,1}(k)}{\lambda + \Phi_{2,1}(k)} \left[ y^*(k+1) - y(k) - \sum_{i=1}^{L_y} \phi_i(k) \Delta y(k - i + 1) \right. \\
- \left. \sum_{i=L_y+1}^{L_u+L_y+1} \phi_i(k) \Delta u(k + L_y - i + 1) \right] 
\]

(8)

According to [1], [2], (8) is also the optimal solution of (9).

Form (4) and (8), we can have

\[
\lambda(1-z^{-1}) [1 - z^{-1}\Phi_{2,1}(z^{-1})] + \Phi_{2,1}(k) \Phi_{2,1}(z^{-1}) y(k+1) 
\]

(10)

The function of the closed-loop poles is

\[
T(z^{-1}) = \lambda(1-z^{-1}) [1 - z^{-1}\Phi_{2,1}(z^{-1})] + \Phi_{2,1}(k) \Phi_{2,1}(z^{-1}) 
\]

(11)

We may place the closed-loop poles in the unit circle to guarantee the system stability by tuning \( \lambda \) and quantitatively analyze the chosen \( \lambda \) based on the locations of closed-loop poles. The steady-state error is typically calculated for linear systems. In this paper, we regard the steady-state error as one property of the transient tendency of the nonlinear systems. It means the tracking error of nonlinear system (1) described by linear system (5) at the time \( k \) tends to. The “steady state error” in the speed (ramp) response of the nonlinear system, as described by the linear system model (5), at time \( k \) is

\[
\lim_{k \to \infty} e(k) = \lim_{z \to 1} - \frac{\frac{1}{\Phi_{2,1}(k)}}{1-z^{-1}J_{1}^{*}(k)}T_{s}z \left[ \frac{1}{1-z^{-1}} \right] 
\]

(12)

where \( T_{s} \) represents the sample time constant. We can conclude that the “steady state error” in the speed response positively relates to \( \lambda \). Specifically, when \( \lambda = 0 \), we will have \( \lim e(k) = 0 \).

This conclusion differs from the theorem that the convergence of tracking error of the system controlled by MFAC is guaranteed under the condition that \( \lambda \) is large enough in [1]-[3]. Besides, the tracking error of step response of the system controlled by the current MFAC converges to zero can be ascribed to that the current MFAC naturally contains one integrator and is unrelated to \( \lambda \).

Furthermore, when system stability can be ensured by setting \( \lambda = 0 \), the static error will be eliminated for a desired trajectory of \( k^n \ (0 \leq n < \infty) \) since

\[
\lim_{k \to \infty} e(k) = \lim_{z \to 1} - \frac{\frac{1}{\Phi_{2,1}(k)}}{1-z^{-1}J_{1}^{*}(k)}C(z) \left[ \frac{1}{1-z^{-1}} \right] 
\]

(13)

where \( Z(k^n) = \frac{C(z)}{(z-1)^{n+1}} \), \( C(z) \) is the polynomial with the highest power of \( n \) and \( Z(\cdot) \) denotes \( z \)-transformation.

**Simulations:**

**Example 1:** The following discrete-time SISO structure-varying linear system is considered in this example.

\[
y(k+1) = \begin{cases} 
-0.4y(k) - 0.5u(k) - 0.6u(k-1) + d, & 1 \leq k \leq 350 \\
0.4y(k) + 0.5u(k) + 0.6u(k-1) + d, & 351 \leq k \leq 700 
\end{cases} 
\]

(14)

where \( d \) is the disturbance. The desired output trajectory is
The controller parameters and initial values for MFAC are listed in Table I. The estimation algorithm adopts the projection algorithm in [1], and [2] with tuning parameters $\eta$ and $\mu$.

| Parameter | MFAC (6) |
|-----------|----------|
| Order     | $L_x = 1$, $L_u = 2$ |
| $\eta$, $\mu$, $\lambda$ | 3; 1; 0.2 |
| Initial value | $\hat{\phi}_1(1) = [-0.1, -0.1, -0.1]$ |
| $u(0:6)$ | $(0, 0, 0, 0, 0, 0)$ |
| $y(0:5)$ | $(0, 0, 0, 0.5, 0.2)$ |

Case 1, $d_1=1$ and $d_2=100$.

Fig. 1 shows the tracking performance of the system controlled by MFAC. Fig. 2 shows the control input. Fig. 3 shows the elements in the estimated PG vector.

From Fig. 1, we can see that the MFAC can remove the influence of constant disturbance on the static error of the system. Because this kind of controller inherently contains one integrator.

Case 2, $d_1=0$ and $d_2=0$.

Fig. 4 shows the tracking performance of the system controlled by MFAC. Fig. 5 shows the elements in the estimated PG vector.

From Fig. 3 and Fig. 5, we can see the sign of the estimated leading coefficient of control input $\hat{\phi}_{ly+1}(k)$ changes at time 350. If the $\hat{\phi}_{ly+1}(k)$ is reset according to [1]-[3], the sign of $\hat{\phi}_{ly+1}(k)$ will become opposite to the actual value after the time of 350. In this way, EDLM can never objectively describe the actual system model, leading to the failure of the controller. Therefore, we allow the estimation algorithm to continue operating in its own way without resetting its value, aiming to validate the fact that all signs of the estimated elements of PG are capable of changing and that the sign of the estimated leading coefficient $\hat{\phi}_{ly+1}(k)$ of the control input should be
altered according to the actual system model. However, the sign of \( \hat{\phi}_{k+1}(k) \) unchanged is an essential precondition for the current stability analysis through the contraction mapping technique. Therefore, it will be more reasonable for us to analyze the stability of the system through the closed-loop function (11) and the static error.

**Example 2**: In this example, the following discrete-time SISO nonlinear system is considered.

\[
y(k+1) = -y^2(k) + u(k) \quad 1 \leq k \leq 201
\]

We choose the desired trajectory with

\[
y^*(k) = -k^2, \quad 1 \leq k \leq 201
\]

to validate the conclusion above about the steady-state error. The MFAC controller (8) is designed with \( \hat{\phi}(k) = 1 \) and

\[
\hat{\phi}(k) = \sum_{i=0}^{\infty} \frac{1}{i!} \frac{\partial^i f(\phi(k-1))}{\partial \phi^i(k-1)} \Delta y^{i-1}(k) = -2y(k-1) - \Delta y(k).
\]

The outputs of the system controlled by MFAC with \( \lambda = 0 \), \( \lambda = 1 \times 10^{-5} \) and \( \lambda = 3 \times 10^{-5} \) are shown in Fig. 6. The calculated PG vector is shown in Fig. 7.

![Fig. 6 Tracking performance](image)

![Fig. 7 Calculated PG vector](image)

From Fig. 6, it is straightforward for someone to see that the static error will increase by raising \( \lambda \). When \( \lambda = 0 \), \( y(200) = y^*(200) = 4 \times 10^{4} \). Furthermore, we can conclude that the tracking error concerning the desired trajectory \( k^n \) ( \( n = 1, 2, \ldots \) ) will not converge to zero until \( \lambda = 0 \). This fact contradicts [1]-[3] which showed that the tracking error of the system controlled by MFAC converges to zero under the condition that \( \lambda \) is sufficiently large.

We change the desired trajectory (16) into

\[
y^*(k) = -k, \quad 1 \leq k \leq 500
\]

and choose \( \lambda = 0.001 \). Define \( e(k) \) as the tracking error at time \( k \) and \( E(k-\text{Inf}) \) as a property of transient tendency, i.e., “steady-state error” of the nonlinear system model at time \( k \) which is calculated by (12), i.e.,

\[
E(k-\text{Inf}) = \lim_{z=1^{-}} \frac{z-1}{\lambda - z} (1 - z^{-1}) \left( 1 - \hat{\phi}(k) \right) + \hat{\phi}(k) (z-1)^2
\]

Fig. 8 shows the tracking performance and calculated PG. Fig. 9 shows the comparison between \( E(k-\text{Inf}) \) and \( e(k) \), \( c(k) = E(k-\text{Inf})/e(k) \) and \( e(k)-E(k-\text{Inf}) \).

![Fig. 8 Tracking performance and Calculated PG](image)

![Fig. 9 Contrast between E(k-Inf) and e(k)](image)

Fig. 9 shows that \( e(k) \) is very close to \( E(k-\text{Inf}) \), confirming our conclusions about transient characteristics.

### III. MODEL-FREE ADAPTIVE CONTROL FOR NONLINEAR SYSTEMS

#### A. System (1) with slight nonlinearity

To explore the application of MFAC in nonlinear systems further, we should begin with Case 2 in the Appendix. The system orders \( n_1+1 \) and \( n_2+1 \) are supposed to be known, and we let \( L_1 = n_1+1 \) and \( L_2 = n_2+1 \). Since \( f(-) \) in (1) is differentiable at the point \( \{y(k-1), \ldots, y(k-n_1-1), u(k-1), \ldots, u(k-n_2-1)\} \), the change in function \( y(k) = f(y(k-1), \ldots, y(k-n_1-1), u(k-1), \ldots, u(k-n_2-1)) \) as the variables change from...
\[ y(k-1), \ldots, y(k-n_y-1), u(k-1), \ldots, u(k-n_u-1) \]

to \[ y(k-1) + dy(k), \ldots, y(k-n_y-1) + dy(k-n_y), u(k-1) + du(k), \ldots, u(k-n_u-1) + du(k-n_u) \] is approximated by the total differential \( dy(k+1) \) as shown in (19).

\[
\Delta y(k+1) \approx dy(k+1) = \frac{\partial f(\phi(k-1))}{\partial y(k-1)} dy(k) + \cdots + \frac{\partial f(\phi(k-1))}{\partial u(k-1)} du(k) + \cdots + \frac{\partial f(\phi(k-1))}{\partial u(k-n_u-1)} du(k-n_u)
\]

where

\[
\Delta H(k) = \begin{bmatrix} \Delta y(k) \\ \Delta y(k-n_y) \\ \Delta u(k) \\ \cdots \\ \Delta u(k-n_u) \end{bmatrix},
\]

\[
\phi(k) = \begin{bmatrix} \phi_1(k) \\ \phi_2(k) \\ \cdots \\ \phi_{n_u}(k) \end{bmatrix},
\]

\[
\phi_{n_u}(k) = \begin{bmatrix} \phi_1(k) \\ \phi_2(k) \\ \cdots \\ \phi_{n_u}(k) \end{bmatrix},
\]

\[
\Delta u(k+1) = \frac{\partial f(\phi(k-1))}{\partial y(k-1)} \Delta y(k) + \cdots + \frac{\partial f(\phi(k-1))}{\partial u(k-1)} \Delta u(k) + \cdots + \frac{\partial f(\phi(k-1))}{\partial u(k-n_u-1)} \Delta u(k-n_u)
\]

The approximation \( \Delta y(k+1) \approx dy(k+1) \) improves as \( dy(k), \ldots, dy(k-n_y), du(k), \ldots, du(k-n_u) \) approach 0. Then we make a further approximation to have model (20), which is also discussed in the Appendix.

\[
\Delta y(k+1) = \begin{bmatrix} \Delta y(k) \\ \Delta y(k-n_y) \\ \Delta u(k) \\ \cdots \\ \Delta u(k-n_u) \end{bmatrix}
\]

where

\[
\Delta H(k) = [\Delta y(k), \ldots, \Delta y(k-n_y), \Delta u(k), \ldots, \Delta u(k-n_u)]^T,
\]

\[
\phi(k) = \begin{bmatrix} \phi_1(k) \\ \phi_2(k) \\ \cdots \\ \phi_{n_u}(k) \end{bmatrix}
\]

\[
= \begin{bmatrix} \phi_1(k) \\ \phi_2(k) \\ \cdots \\ \phi_{n_u}(k) \end{bmatrix}
\]

When the control period of the system is sufficiently small, the model (20) accurately describes the system. Similarly to Section II, the MFAC controller is still written as (8). On the other hand, we can design the MFAC controller by minimizing the quadratic function (9) subject to constraint \( u_m(k) \leq u(k) \leq u_M(k) \), similar to the following B part.

**B. System (1) with strong nonlinearity**

i) If the system is strongly nonlinear and the control period is not sufficiently small, the obtained \( \phi(k) \) may have an apparent change from time \( k = 1 \) to \( k+1 \), sometimes leading to poor system performance. To this end, we recommend applying the iterative MFAC controller, as suggested in [8] and [9]. The controller will be

\[
\Delta u(k+1) = \frac{\partial f(\phi(k-1))}{\partial y(k-1)} \Delta y(k) + \cdots + \frac{\partial f(\phi(k-1))}{\partial u(k-1)} \Delta u(k) + \cdots + \frac{\partial f(\phi(k-1))}{\partial u(k-n_u-1)} \Delta u(k-n_u)
\]

where \( i \) is the iteration number before the control input is sent to the system at \( k+1 \).

ii) On the other hand, if the function \( f(\cdot) \) has derivatives of all orders at any operating points, the system model (1) can be described as (22) by the Appendix.

\[
\Delta y(k+1) = \frac{\partial f(\phi(k-1))}{\partial y(k-1)} \Delta y(k) + \cdots + \frac{\partial f(\phi(k-1))}{\partial u(k-1)} \Delta u(k) + \cdots + \frac{\partial f(\phi(k-1))}{\partial u(k-n_u-1)} \Delta u(k-n_u)
\]

where \( \gamma(k) \) is defined in the Appendix. Let

\[
\phi(k) = \begin{bmatrix} \frac{\partial f(\phi(k-1))}{\partial y(k-1)} + e_1(k), \ldots, \frac{\partial f(\phi(k-1))}{\partial y(k-n_y-1)} + e_{n_y}(k), \\
\frac{\partial f(\phi(k-1))}{\partial u(k-1)} + e_{n_y+1}(k), \ldots, \frac{\partial f(\phi(k-1))}{\partial u(k-n_u-1)} + e_{n_u+n_u}(k) \end{bmatrix}
\]

\[
e_{n_y+1}(k) = \frac{1}{2!} \frac{\partial^2 f(\phi(k-1))}{\partial y^2(k-i-1)} \Delta y(k-i) + \frac{1}{3!} \frac{\partial^3 f(\phi(k-1))}{\partial y^3(k-i-1)} \Delta y^3(k-i)
\]

\[
e_{n_u+n_u+1}(k) = \frac{1}{2!} \frac{\partial^2 f(\phi(k-1))}{\partial u^2(k-j-1)} \Delta u(k-j) + \frac{1}{3!} \frac{\partial^3 f(\phi(k-1))}{\partial u^3(k-j-1)} \Delta u^3(k-j) + \cdots
\]

\[
\frac{1}{n!} \frac{\partial^n f(\phi(k-1))}{\partial u^n(k-1)} \Delta u^n(k-1)
\]

\[
\Delta u^i(k) = \frac{1}{2!} \frac{\partial^2 f(\phi(k-1))}{\partial u^2(k-j-1)} \Delta u(k-j) + \frac{1}{3!} \frac{\partial^3 f(\phi(k-1))}{\partial u^3(k-j-1)} \Delta u^3(k-j) + \cdots
\]

\[
\frac{1}{n!} \frac{\partial^n f(\phi(k-1))}{\partial u^n(k-1)} \Delta u^n(k-1)
\]

\[
, i = 0, \ldots, n_y, j = 0, \ldots, n_u, \text{ where (24) and (25) are collected from (41) in Appendix, and then (22) is rewritten as (4).}
\]

a) If \( \frac{\partial f(\phi(k-1))}{\partial u(k-1)} \) is a nonzero constant, the controller is still described by (8) and the closed-loop system equation at time \( k \) is still (10).

b) If there exists \( \frac{1}{n!} \frac{\partial^n f(\phi(k-1))}{\partial u^n(k-1)} \neq 0 \) for \( n \geq 1 \) and

\[
\frac{1}{j!} \frac{\partial^j f(\phi(k-1))}{\partial u^j(k-1)} = 0 \quad (j > n),
\]

we may obtain the control law by minimizing the following cost function of at least \( 2n \)-th degree (26) or by minimizing the cost function (27) subject to constraint \( u_m(k) \leq u(k) \leq u_M(k) \).
\[ J = \left[ y'(k+1) - y(k+1) \right]^2 + \lambda \left| \Delta u(k) \right|^2 \]

\[ = \left[ y'(k+1) - y(k) - \sum_{i=1}^{K} \phi_i(k) \Delta y(k-i+1) \right]^2 + \lambda \left| \Delta u(k) \right|^2 \]  

\[ \min_{u(k) \in \mathbb{U}(1) \Delta u(k) \in \mathbb{U}(1)} J = \left[ y'(k+1) - y(k+1) \right]^2 + \lambda \left| \Delta u(k) \right|^2 \]  

Besides, to obtain the controller shown as (8) or to simplify the minimization of (27), we may simplify the controller design process through an approximation:
\[ \phi_{L+1}(k) \approx \frac{\partial f(\phi(k-1))}{\partial u(k-1)} + \frac{1}{2!} \frac{\partial^2 f(\phi(k-1))}{\partial^2 u^2(k-1)} \Delta u(k-1) \]
\[ + \frac{1}{3!} \frac{\partial^3 f(\phi(k-1))}{\partial^3 u^3(k-1)} \Delta u^3(k-1) + \ldots \]  

where the coefficient (28) stems from (41) and \( \Delta u(k) \) is replaced by \( \Delta u(k-1) \).

**Simulations:**

**Example 3:** The following discrete-time SISO nonlinear system is considered.
\[ y(k+1) = 0.2 y^2(k) + 2u(k) + 2u^2(k) + 2u^3(k-1) + \cos(u(k-1)) + u^6(k-2) \]  

The desired output trajectory is
\[ y'(k) = \begin{cases} 
0.5 \sin(k/50) + 0.5 \cos(k/3) & 1 \leq k \leq 350 \\
+ 0.5 \sin(k/10) & 351 \leq k \leq 700 \\
0.3 + 0.3 \times (-1)^{\text{round}(k/50)} & 351 \leq k \leq 700 
\end{cases} \]  

The controller parameters and initial values for MFAC are listed in Table II.

| Parameter Settings for MFAC | MFAC (6) |
|-----------------------------|----------|
| Order | \( L_{\text{in}}=1, L_{\text{out}}=2 \) |
| Initial value | \( \lambda = 1.5 \) |
| \( \lambda \) | \( \hat{\phi}_k(3) = \hat{\phi}_k(2) = \hat{\phi}_k(1) = [0.01, 0.01, 0.01, 0.01] \) |
| \( y(0:4) \) | \( u(0:3) = 0 \) |

The controller MFAC 1 is designed by minimizing a quartic equation (26) shown as
\[ J = \left[ y'(k+1) - y(k+1) \right]^2 + \lambda \left| \Delta u(k) \right|^2 \]
\[ = \left[ y'(k+1) - y(k) - \hat{\phi}_k(k) \Delta y(k) - \hat{\phi}_k(k) \Delta u(k-1) - \hat{\phi}_k(k) \Delta u(k-2) \right]^2 + \lambda \left| \Delta u(k) \right|^2 \]
\[ = \frac{\partial f(\phi(k-1))}{\partial u(k-1)} \Delta u(k-1) - \frac{1}{2!} \frac{\partial^2 f(\phi(k-1))}{\partial^2 u^2(k-1)} \Delta u^2(k-1) + \ldots \]

with \( \hat{\phi}_k(k) = \sum_{i=1}^{K} \frac{1}{i!} \frac{\partial^i f(\phi(k-1))}{\partial^i u^i(k-1)} \Delta y^{i-1}(k) \),

\[ \hat{\phi}_k(k) = \sum_{i=1}^{K} \frac{1}{i!} \frac{\partial^i f(\phi(k-1))}{\partial^i u^i(k-1)} \Delta u^{i-1}(k) \]

\[ \hat{\phi}_k(k) = \sum_{i=1}^{K} \frac{1}{i!} \frac{\partial^i f(\phi(k-1))}{\partial^i u^i(k-1)} \Delta u^{i-1}(k-1) \]

We apply the controller (8) named MFAC 2. \( \hat{\phi}_k(k) \) and \( \hat{\phi}_k(k) \) are the same as those of MFAC 1 and \( \hat{\phi}_k(k) \) is approximated by (28): \( \hat{\phi}_k(k) = 2 + 2u(k-1) + \Delta u(k-1) \).

The controller MFAC 3 is designed to minimize (24) subject to the control input constraint \(-0.6 \leq u(k) \leq -0.2 \)

![Fig. 10 Tracking performance](image1.png)

![Fig. 11 Control input](image2.png)

![Fig. 12 Elements in calculated PG vector](image3.png)

**Question:** Why don’t we use the MFAC by introducing an online estimation algorithm like [1]?
Answer: Because the model can hardly be established in this way. One can refer to the Remark in [7] for detailed explanations.

IV. CONCLUSION

In this note, the stability of the system and the chosen parameter λ are analyzed by the closed-loop function, and we have figured out that the current MFAC methods are not analyzed correctly. Then, several simulated examples are used to validate the viewpoints. According to its nature, we should rename MFAC as the incremental one-step-ahead optimal controller.

APPENDIX

Proof of Theorem 1

From (1), we have

\[ \Delta v(k + 1) = f(y(k), \ldots, y(k - L_y + 1), y(k - L_y), \ldots, y(k - n_y)), u(k), \ldots, u(k - L_u + 1), u(k - L_u), \ldots, u(k - n_u)) \]

\[ - f(x(k - 1), y(k - L_y), y(k - L_y), \ldots, y(k - n_y), u(k - 1), \ldots, u(k - L_u), u(k - L_u), \ldots, u(k - n_u)) \]

\[ + f(y(k - 1), \ldots, y(k - L_y), y(k - L_y), \ldots, y(k - n_y), u(k - 1), \ldots, u(k - L_u), u(k - L_u), \ldots, u(k - n_u)) \]

\[ - f(y(k - 1), \ldots, y(k - L_y), y(k - L_y), \ldots, y(k - n_y), u(k - 1), \ldots, u(k - L_u), u(k - L_u), \ldots, u(k - n_u)) \]

\[ - f(y(k - 1), \ldots, y(k - L_y), u(k - 1), \ldots, y(k - n_y), u(k - 1), \ldots, u(k - L_u), u(k - L_u), \ldots, u(k - n_u)) \]

\[ + f(y(k - 1), \ldots, y(k - L_y), u(k - 1), \ldots, y(k - n_y), u(k - 1), \ldots, u(k - L_u), u(k - L_u), \ldots, u(k - n_u)) \]

(32)

On the basis of Assumption 1 and the definition of differentiability in (15), (32) becomes

\[ \Delta v(k + 1) = \frac{\partial f \phi(k - 1)}{\partial y(k - 1)} \Delta y(k) + \cdots + \frac{\partial f \phi(k - 1)}{\partial y(k - L_y)} \Delta y(k - L_y + 1) \]

\[ + \frac{\partial f \phi(k - 1)}{\partial u(k - 1)} \Delta u(k) + \cdots + \frac{\partial f \phi(k - 1)}{\partial u(k - L_u)} \Delta u(k - L_u + 1) \]

\[ + \epsilon_1(k) \Delta Y(k) + \cdots + \epsilon_{L_y}(k) \Delta Y(k - L_y + 1) + \epsilon_{L_y + 1}(k) \Delta Y(k) \]

\[ + \cdots + \epsilon_{L_y + L_u}(k) \Delta Y(k - L_u + 1) + \psi(k) \]

(33)

where

\[ \psi(k) = f(y(k - 1), \ldots, y(k - L_y), u(k - L_u), \ldots, u(k - L_u)) \]

\[ - f(y(k - 1), \ldots, y(k - L_y), u(k - L_u), \ldots, u(k - L_u)) \]

\[ + f(y(k - 1), \ldots, y(k - L_y), u(k - L_u), \ldots, u(k - L_u)) \]

\[ - f(y(k - 1), \ldots, y(k - L_y), u(k - L_u), \ldots, u(k - L_u)) \]

\[ - \epsilon_1(k) \Delta Y(k) - \cdots - \epsilon_{L_y}(k) \Delta Y(k - L_y + 1) - \epsilon_{L_y + 1}(k) \Delta Y(k) \]

\[ + \cdots + \epsilon_{L_y + L_u}(k) \Delta Y(k - L_u + 1) + \psi(k) \]

(34)

denote the partial derivative values of \( f(\phi(k - 1)) \) with respect to the \((i+1)\)-th variable and the \((n_i+2j)-th\) variable, respectively. And \( \epsilon_1(k), \ldots, \epsilon_{L_y + L_u}(k) \) are functions that depend only on \( \Delta y(k), \Delta y(k - L_y + 1), \Delta y(k - L_y), \ldots, \Delta u(k - L_u + 1) \), with \( \epsilon_1(k), \ldots, \epsilon_{L_y + L_u}(k) \rightarrow (0, \ldots, 0) \) when \( \Delta y(k), \Delta y(k - L_y + 1), \Delta y(k - L_y), \ldots, \Delta u(k - L_u + 1) \rightarrow (0, \ldots, 0) \). This also implies that \( (\epsilon_1(k), \ldots, \epsilon_{L_y + L_u}(k)) \) can be regarded as \((0, \ldots, 0)\) when the control period of the system is sufficiently small.

We consider the following equation with the vector \( \eta(k) \) for each time \( k \):

\[ \psi(k) = \eta^T(k) \Delta H(k) \]

(35)

Owing to \( \| \Delta H(k) \| \neq 0 \), (35) must have at least one solution \( \eta(k) \). Let

\[ \phi(k) = \eta^T(k) + [\frac{\partial f \phi(k - 1)}{\partial y(k - 1)} + \epsilon_1(k), \ldots, \frac{\partial f \phi(k - 1)}{\partial y(k - L_y)} + \epsilon_{L_y}(k), \]

\[ + \frac{\partial f \phi(k - 1)}{\partial u(k - 1)} + \epsilon_{L_y + 1}(k), \ldots, \frac{\partial f \phi(k - 1)}{\partial u(k - L_u)} + \epsilon_{L_y + L_u}(k)]^T \]

(36)

(33) can be described as follows:

\[ \Delta y(k + 1) = \phi(k) \]

(37)

Case 2: \( L_y = n_y + 1 \) and \( L_u = n_u + 1 \)

On the basis of Assumption 1 and the definition of differentiability in (15), (1) becomes

\[ \Delta y(k + 1) = \frac{\partial f \phi(k - 1)}{\partial y(k - 1)} \Delta y(k) + \cdots + \frac{\partial f \phi(k - 1)}{\partial y(k - L_y)} \Delta y(k - L_y + 1) \]

\[ + \frac{\partial f \phi(k - 1)}{\partial u(k - 1)} \Delta u(k) + \cdots + \frac{\partial f \phi(k - 1)}{\partial u(k - L_u)} \Delta u(k - L_u + 1) \]

\[ + \gamma(k) \]

(38)

where

\[ \gamma(k) = \epsilon_1(k) \Delta y(k) + \cdots + \epsilon_{L_y}(k) \Delta y(k - n_y) \]

\[ + \epsilon_{L_y + 1}(k) \Delta u(k) + \cdots + \epsilon_{L_y + L_u}(k) \Delta u(k - n_u) \]

(39)

We let

\[ \phi(k) = \left[ \frac{\partial f \phi(k - 1)}{\partial y(k - 1)} + \epsilon_1(k), \ldots, \frac{\partial f \phi(k - 1)}{\partial y(k - L_y)} + \epsilon_{L_y}(k), \right. \]

\[ + \frac{\partial f \phi(k - 1)}{\partial u(k - 1)} + \epsilon_{L_y + 1}(k), \ldots, \frac{\partial f \phi(k - 1)}{\partial u(k - L_u)} + \epsilon_{L_y + L_u}(k)]^T \]

(40)

to rewrite (38) as (37), with \( (\epsilon_1(k), \ldots, \epsilon_{L_y + L_u}(k)) \rightarrow (0, \ldots, 0) \) in nonlinear systems, when \( (\Delta y(k), \ldots, \Delta y(k - n_y), \Delta u(k), \ldots, \Delta u(k - n_u)) \rightarrow (0, \ldots, 0) \). As to linear systems, we will always have

\[ \phi(k) = \left[ \frac{\partial f \phi(k - 1)}{\partial y(k - 1)} \right. \]

\[ + \frac{\partial f \phi(k - 1)}{\partial y(k - n_y - 1)} \frac{\partial f \phi(k - 1)}{\partial u(k - 1)} \]

\[ + \frac{\partial f \phi(k - 1)}{\partial u(k - n_u - 1)} \right] \]

(41)

no matter what \( (\Delta y(k), \ldots, \Delta y(k - L_y), \Delta u(k), \ldots, \Delta u(k - L_u)) \) is.

Additionally, if the function \( f(\cdot) \) has derivatives of all orders on any operating points, we can obtain (41) by the Taylor series.
\[ \Delta y(k+1) = [\Delta y(k) \frac{\partial}{\partial y(k-1)} + \ldots + \Delta y(k-n_y) \frac{\partial}{\partial y(k-n_y-1)} + \Delta u(k) \frac{\partial}{\partial u(k-1)} + \ldots + \Delta u(k-n_u) \frac{\partial}{\partial u(k-n_u-1)}]f(\varphi(k-1)) + \frac{1}{2!} [\Delta y(k) \frac{\partial}{\partial y(k-1)} + \ldots + \Delta y(k-n_y) \frac{\partial}{\partial y(k-n_y-1)} + \Delta u(k) \frac{\partial}{\partial u(k-1)} + \ldots + \Delta u(k-n_u) \frac{\partial}{\partial u(k-n_u-1)}]^2 f(\varphi(k-1)) + \ldots \]

and then obtain a group of solutions (42), (43) for (39) as follows
\[ \varepsilon_{i+1}(k) = \frac{1}{2!} \frac{\partial^2 f(\varphi(k-1))}{\partial y^2(i-1)} \Delta y(k-i) + \frac{1}{3!} \frac{\partial^3 f(\varphi(k-1))}{\partial y^3(i-1)} \Delta y^3(k-i) + \ldots \]
\[ \varepsilon_{i,j+1}(k) = \frac{1}{2!} \frac{\partial^2 f(\varphi(k-1))}{\partial y^2(i-j)} \Delta u(k-j) + \frac{1}{3!} \frac{\partial^3 f(\varphi(k-1))}{\partial y^3(i-j)} \Delta u^3(k-j) + \ldots \]

Case 3: \( L_u > n_u + 1 \) and \( L_y > n_y + 1 \)

Based on Assumption 1 and the definition of differentiability in [15], (1) becomes

\[ \Delta y(k+1) = \frac{\partial f(\varphi(k-1))}{\partial y(k-1)} \Delta y(k) + \ldots + \frac{\partial f(\varphi(k-1))}{\partial y(k-n_y)} \Delta y(k-n_y) + \frac{\partial f(\varphi(k-1))}{\partial u(k-1)} \Delta u(k) + \ldots + \frac{\partial f(\varphi(k-1))}{\partial u(k-n_u)} \Delta u(k-n_u) + e_k(k) \Delta y(k) + e_{i+1}(k) \Delta u(k-n_u) + \ldots + e_{i,n_u+1}(k) \Delta u(k-n_u) \]

Define
\[ \gamma(k) = \varepsilon_k(k) \Delta y(k) + \ldots + e_{i,n_y+1} k(k) \Delta y(k-n_y) + e_{i+1}(k) \Delta u(k-n_u) + \ldots + e_{i,n_u+1}(k) \Delta u(k-n_u) \]

We consider the following equation with the vector \( \eta(k) \) for each time \( k \):
\[ \gamma(k) = \eta^T(k) \Delta H(k) \]

Owing to \( \|\Delta H(k)\| \neq 0 \), (46) must have at least one solution \( \eta(k) \).

Let
\[ \phi_k(k) = \eta^T(k) + \frac{\partial f(\varphi(k-1))}{\partial y(k-1)} \Delta y(k-1) + \ldots + \frac{\partial f(\varphi(k-1))}{\partial y(k-n_y)} \Delta y(k-n_y) + \ldots + \frac{\partial f(\varphi(k-1))}{\partial u(k-1)} \Delta u(k-1) + \ldots + \frac{\partial f(\varphi(k-1))}{\partial u(k-n_u)} \Delta u(k-n_u) + \ldots \]

Then (44) can be rewritten as (37).

Case 4: \( L_u \geq n_u + 1 \) and \( 1 \leq L_y \leq n_y + 1 \) and \( L_u \geq n_u + 1 \) and \( L_y \leq n_u + 1 \).

The proof of Case 4 is similar to the above analysis process, we omit it.

We finished the proof of Theorem 1.

Remark 2: The UD in Case 2 (\( L_u = n_u + 1 \) and \( L_y = n_u + 1 \)) is shown as follows.
\[ y(k) = \frac{\partial f(\varphi(k-1))}{\partial y(k-1)} y(k-1) + \ldots + \frac{\partial f(\varphi(k-1))}{\partial y(k-n_y)} y(k-n_y) \] 
\[ + \frac{\partial f(\varphi(k-1))}{\partial u(k-1)} u(k-1) + \ldots + \frac{\partial f(\varphi(k-1))}{\partial u(k-n_u)} u(k-n_u) + v(k) \]

where (49) represents the UD around the operating point, according to the conception in [16].

\[ v(k) = y(k+1) - \frac{\partial f(\varphi(k-1))}{\partial y(k-1)} y(k) - \ldots - \frac{\partial f(\varphi(k-1))}{\partial y(k-n_y)} y(k) - v(k) - \frac{\partial f(\varphi(k-1))}{\partial u(k-1)} u(k) - \ldots - \frac{\partial f(\varphi(k-1))}{\partial u(k-n_u)} u(k-n_u) - y(k) \]

(49)

REFERENCES

[1] Hou Z S, Xiong S S. On Model Free Adaptive Control and its Stability Analysis[J]. IEEE Transactions on Automatic Control, 2019, 64(11): 4555-4569.

[2] Hou Z S and Jin S T. Model Free Adaptive Control: Theory and Applications, CRC Press, Taylor and Francis Group, 2013.

[3] Hou Z S and Jin S T, “A novel data-driven control approach for a class of discrete-time nonlinear systems,” IEEE Transaction on Control Systems Technology, vol. 19, no. 6, pp. 1549-1558, 2011.

[4] Goodwin G C, Sin K S. Adaptive Filtering Prediction and Control[M]. Dover Publications, Inc, 2009.

[5] Åström K J, Wittenmark B. Adaptive control [M]. Courier Corporation, 2013.

[6] Fu Y, Chai T Y. Nonlinear multivariable adaptive control using multiple models and neural networks[J]. Automatica, 2007, 43(6): 1101-1110.

[7] Zhang F L, “A New Model-Free Method for MIMO Systems and Discussion on Model-Free or Model-Based[J].” 2020, arXiv:2007.02761. [Online]. Available: https://arxiv.org/abs/2007.02761

[8] Zhang F L. Discussion on a Class of Model-Free Adaptive Control for Multivariable Systems[J]. 2020, arXiv:2010.14847. [Online]. Available: https://arxiv.org/abs/2010.14847

[9] Zhang F L. Discussions on Inverse Kinematics based on Levenberg-Marquardt Method and Model-Free Adaptive (Predictive) Control[J]. 2020, arXiv:2009.14507. [Online]. Available: https://arxiv.org/abs/2009.14507

[10] Zhang Y J, Chai T Y, et al. Data-Driven PID Controller and Its Application to Pulp Neutralization Process[J]. IEEE Transactions on Control Systems Technology, 2018, 26(3):828-841.

[11] Zhang Y J, Chai T Y, Wang H, et al. Nonlinear Decoupling Control With ANFIS-Based Unmodeled Dynamics Compensation for a Class of Complex Industrial Processes[J]. IEEE Transactions on Neural Networks & Learning Systems, 2018, 29(99):2352-2366.

[12] Zhang Y J, Chai T, Wang D H, et al. Virtual Unmodeled Dynamics Modeling for Nonlinear Multivariable Adaptive Control With Decoupling Design[J]. IEEE Transactions on Systems Man & Cybernetics Systems, 2016:1-12.

[13] Zhang Y J, Chai T Y, Wang H. A Nonlinear Control Method Based on ANFIS and Multiple Models for a Class of SISO Nonlinear Systems and Its Application[J]. IEEE Transactions on Neural Networks, 2011, 22(11):1783-1795.

[14] Zhang Y J, Chai T Y, Wang H, et al. An Improved Estimation Method for Unmodeled Dynamics Based on ANFIS and Its Application to Controller Design[J]. IEEE Transactions on Fuzzy Systems, 2013, 21(6):989-1005.

[15] Briggs W, Cochran L, Gillett B. Calculus [M]. Pearson, 2014.

[16] Chai T Y, Yue H. Adaptive Control [M]. Beijing: Tsinghua University Press, 2015.